Министерство образования и науки Российской Федерации
ФГБОУ ВО "Уральский государственный педагогический университет"
Институт физики, технологии и экономики
Кафедра физики и математического моделирования

«Анализ и использование инструментов программирования для операционной системы Android»

Выпускная квалификационная работа
Направление: 09.03.03 - Прикладная информатика
Профиль: Прикладная информатика в сервисе

Квалификационная работа
допущена к защите
Заведующий кафедрой
д.ф-м.н., профессор
Сидоров Валерий Евгеньевич

_____ _____
дата подпись

Исполнитель:
Харламов Аркадий
Андреевич
Группа БП-42

_____
подпись

Научный руководитель:
к.п.н., доцент
Стихина Наталья
Владимировна

_____
подпись

Екатеринбург 2016

Оглавление

Введение

В настоящее время компьютерная техника стала неотъемлемой частью жизни человека. С прогрессом в технологиях компьютерная техника развиваясь, так же и меняла свой вид и размеры. Появились портативные устройства, которые человек может переносить с собой на протяжении дня и использовать в любом месте и в любое время. По функциональности эти устройства близки к персональным компьютерам. Они образуют такой класс компьютерной техники, как мобильные устройства. Примерами таких устройств используемых человеком в повседневной жизни является мобильный телефон (смартфон), планшетный компьютер, умные часы и т.п. Эти устройства используются человеком в качестве помощников. Они обеспечивают доступ к различным мировым информационным ресурсам, позволяют общаться с другими людьми, их использовать для досуга и многое другое. В связи с ростом технологий и возможностей, мобильные устройства имеют очень большой спрос в настоящее время.

Важной часть мобильного устройства является операционная система. Одна из операционных систем, используемых в мобильных устройствах, называется Android. По количеству пользователей, Android является лидером среди операционных систем используемых в мобильной технике.

В мобильных устройствах в качестве программного обеспечения используются мобильные приложения. Мобильные приложения устанавливаются на мобильное устройство с различных площадок. Самой популярной площадкой в операционной системе Android является Google Play Market. При помощи Google Play Market разработчики выкладывают свои приложения, а пользователи могут их приобретать, скачивать и устанавливать на свое мобильное устройство. Мобильные приложения могут нести в себе различный функционал. Приложения, в зависимости от своего назначения, являются инструментом реализации потребностей человека в мобильном устройстве.

Исходя из выше сказанного, становится понятно, что разработка приложений для мобильных устройств на операционной системе Android является очень перспективной деятельностью.

Целью данной выпускной квалификационной работы является разработка приложения, функционирующее на платформе Android, которое будет помогать студентам Института физики, технологии и экономики УрГПУ получать информацию о своем факультете.

Задачами данной выпускной квалификационной работы являются:

1) анализ существующих решений;

2) анализ инструментов, необходимых для реализации цели;

3) анализ механизмов, необходимых для осуществления цели;

4) разработка приложения.

Глава 1. Теоретическая часть

1.1 История операционной системы Android

История создания ОС Android началась в октябре 2003 года. В то время, в Palo Alto, Клифорния, США, была основана небольшая организация под названием Android Incorporated. Её учредителями были Рич Майнер, Крис Уайт, Энди Рубин и Ник Сирс. Целью создания компании являлось создание операционной системы для мобильных устройств, которые с ее помощью были бы хорошо осведомлены о местоположении и предпочтениях его владельца. Несмотря на прошлые достижения основателей и ранних сотрудников, Android Inc работала тайно, анонсируя только то, что трудится над новой платформой для мобильных телефонов. В том же году у Рубина закончились деньги и Стив Перлман, близкий друг Рубина, одолжил ему $10.000 наличными и таким образом получил долю в компании. В связи с работой в условиях особой секретности, у компании не было инвесторов и без возможности дальнейшего финансирования разработок, история операционной системы Android могла бы закончиться. Благодаря аналитикам компании Google Android получил второе рождение.

Google приобрел Android Incorporated 17 августа 2005 года и сделал его своей дочерней компанией. Ключевые сотрудники Android Incorporated, в том числе Рубин, Мэйнер и Вайт остались в компании после слияния. Несмотря на то, что информации об Android Incorporated было довольно мало, многие прозорливые аналитики высказали предположение, что Google таким маневром собирается обосноваться на рынке мобильных устройств. В составе Google команда разработчиков во главе с Рубином завершила создание инновационной операционной системы для мобильных устройств на базе ядра Linux. Платформа была презентована производителям электроники и сотовым операторам как гибкая, модифицируемая и легко обновляемая система.

Слухи о намерении компании Google обосноваться на рынке мобильной связи с так называемым «Гуглфоном» стали усиленно распространяться с конца 2006 года. Печатные и интернет СМИ сообщали, что Google разрабатывает телефон под собственной торговой маркой. Некоторые даже утверждали, что Google уже провел закрытую презентацию своего инновационного продукта сотовым операторам и производителям смартфонов. В сентябре 2007 года появляется информация, что Google отправила ряд заявок на патенты в сфере мобильной связи.

5 ноября 2007 года был образован Open Handset Alliance - техноальянс из 48-х крупнейших корпораций, состоящий из производителей телефонов, компьютеров, чипсетов, программного обеспечения и операторов сотовой связи, основной целью создания которого стал переход рынка мобильных устройств на открытые стандарты. В альянс входят такие компании как Google, HTC, Broadcom Corporation, Qualcomm, Intel, LG, Motorola, Samsung Electronics, T-Mobile и другие. В тот же день, в качестве своего первого продукта был презентован Android 1.0, мобильная платформа для сенсорных смартфонов, построенная на ядре Linux 2.6.

С 2008 года платформа Android сильно эволюционировала, пройдя через многочисленные обновления, которые постепенно улучшали операционную систему, расширяли возможности и исправляли ошибки предыдущих версий. Каждый крупный релиз носит название десерта или кондитерской сладости, первые буквы которого идут в соответствии с латинским алфавитом, например: после версии 1.0 Apple Pie шла версия 1.1 Banana Bread, затем 1.5 Cupcake и 1.6 Donut. Последняя представленная в конце 2015 году версия Android 6.0 носит название Muffin (в переводе с английского - оладья). На рисунке 1 представлена диаграмма, показывающая глобальное использование разных версий Android с 2009 по 2016 год.

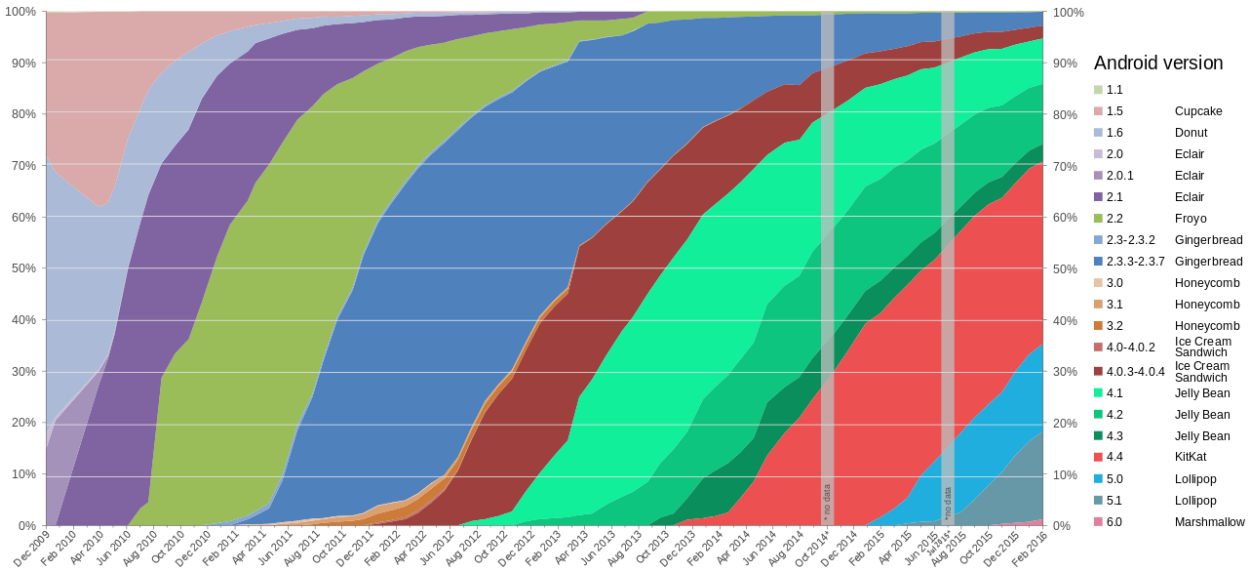

Рис. 1. Статистика Google по версиям ОС Android и их популярности среди пользователей

Версии операционной системы Android:

- Android 1.0 Apple Pie (23 сентября 2008 года).

Это была тестовая платформа Android. Оценить ее возможности можно было, приобретя первый "гуглофон" - HTC Dream, который, кстати, сразу же окрестили главным конкурентом iPhone. И, в общем, было за что, функционально HTC был богаче. В первую очередь, здесь была многозадачность. Из стандартных программ имелись GPS, Bluetooth, Google Maps, Gmail, YouTube и т.д. Android Market тоже имелся, однако насчитывал всего 35 приложений.

- Android 1.1 Banana Bread (9 февраля 2009 года).

Это было небольшое обновление, которое выпустили специально для HTC Dream. Оно включало ряд исправлений ошибок, которых в тестовой версии имелось предостаточно. В меню появились кнопки "Show" и "Hide", кроме того, стало возможным сохранять вложения MMS.

- Android 1.5 Cupcake (30 апреля 2009 года).

В этой версии была масса изменений и доработок. Во-первых, были исправлены ошибки прежних версий. Во-вторых, весомо улучшена стабильность и повышена скорость работы. Появилась новая клавиатура с функцией автозаполнения и работы при различных ориентациях экрана. Кроме того, пользователю стали доступны два Bluetooth-профиля: A2DP и AVRCP, а

также запись и воспроизведение видеороликов в форматах MPEG-4 и 3GP. Приятным дополнением стала поддержка виджетов и папок на рабочем столе.

- Android 1.6 Donut (15 сентября 2009 года).

Обновленная версия приобрела много интересных функциональных возможностей. В частности гораздо удобнее стала функция поиска, теперь пользователь мог вести поиск по истории, закладкам, контактам и в интернете. Кроме того, добавился мультиязычный голосовой поиск, большое внимание было уделено более комфортной работе с Android Market.

- Android 2.0/2.1 Eclair (26 октября 2009 года).

С этой версией появилась возможность использования нескольких аккаунтов Google, весомо изменился пользовательский интерфейс браузера, а также была внедрена поддержка HTML5. Камера наконец-то приобрела вспышку, цифровой зум и функцию макросъемки. Появилась поддержка новых размеров и разрешений экрана, улучшена раскладка электронной клавиатуры и пользовательский интерфейс.

- Android 2.2 Froyo (20 мая 2010 года).

Версия принесла массу существенных изменений. Была проделана большая работа по общей оптимизации, появилась поддержка Adobe Flash 10.1, были ведены Wi-Fi hotspot и Tethering. Появилась возможность установки приложений на карту памяти и поддержка экранов со сверхвысоким разрешением.

- Android 2.3 Gingerbread (6 декабря 2010 года).

В этой версии обновился дизайн пользовательского интерфейса, была улучшена программная клавиатура, введена поддержка большего числа сенсоров, а также сверхвысоких размеров экранов и разрешений. Ощутимые улучшения коснулись возможностей управления питанием и контроля за приложениями.

- Android 3.0 Honeycomb (22 февраля 2011 года).

Это версия Android предназначалась специально для планшетных компьютеров, а потому большое внимание было уделено повышению качества многозадачности. Появился трехмерный рабочий стол, была проведена полноценная оптимизация для устройств с большими экранами.

- Android 4.0 ICS Ice Cream Sandwich (19 октября 2011 года).

Представление данной версией стало своего рода революционным событием в хронологии версий Android. Ice Cream Sandwich стала единой платформой для смартфонов и для планшетов. Из наиболее значимых нововведений можно отметить появление поддержки Wi-Fi Direct и Real-time Transport Protocol API, возможность создания папок на рабочем столе и реализации функции "screen capture" средствами операционной системы, систему контроля использования интернет-трафика, программные улучшения камеры и обновленный браузер с новыми возможностями.

- Android 4.1/4.2/4.3 Jelly Bean (27 июня 2012 года).

Версия сделала систему более производительной, а интерфейс стал плавнее, за счет использования технологии Project Butter. Кроме того, появились возможности голосового ввода в автономном режиме и сохранения фотографий контактов в хорошем качестве, обновленная виртуальная клавиатура с системой предиктивного ввода, расширенным словарем с поддержкой новых языков, автоматическое масштабирование виджетов, более широкая панель уведомлений, сервис Google Now, собирающий полезную информацию для пользователя, опираясь на историю запросов поиска и календарь.

29 октября 2012 года была представлена Android 4.2, она не принесла значительных изменений, а потому название осталось прежним Jelly Bean. Главным новшеством можно назвать возможность создания сразу нескольких учетных записей на одной устройстве, в том случае, когда его использует несколько людей. Таким образом, каждый пользователь мог создать комфортный для себя профиль со своими настройками. Для клавиатуры появилась поддержка функции ввода жестами, настройки камеры получили новый режим: Photo Sphere, позволяющий создавать панорамы с углом обзора до 360

градусов. Кроме того появились: быстрый доступ к настройкам, виджеты на экране блокировки, Google Now стал использовать Gmail как источник данных, появилась функция Daydream для демонстрации заставок при подключении устройства к док-станции и возможность беспроводной передачи видео и игр на совместимые телевизоры.

4.3 представлена 24 июля 2013 года. В новой версии увеличена производительность, за счет оптимизации в работе с многоядерностью. Время переключение задач также стало меньше. 3 версия оснащена поддержкой Bluetooth AVRCP 1.3 для управления медиа-приложениями. Теперь система передает кроме команд еще и данные об исполнителе. Появилась возможность создания нескольких профилей, которые изолированы друг от друга и имеют разграниченные файлы и настройки. В Android 4.3 повышен API до 18 версии. Местоположение определяется на аппаратном уровне, что экономит заряд батареи. Также при выборе постоянной работы Wi-Fi возможно улучшенное определение геолокации. Данные с сенсора теперь определяются без использования магнитного поля. Разработана новая функция для видеоплеера в которой звук и видео соединяются из разных файлов во время воспроизведения. Поддержка стандарта OpenGL ES 3.0. Разработчикам стало доступно больше возможностей видеочипов.

- 4.4 KitKat (31 октября 2013).

Изменения более глобальные, особенно все стало очень удобно для разработчиков приложений. Для бюджетных смартфонов оптимизирована работа. Голосовой помощник стал активен постоянно, можно в любой момент отдавать смартфону команды. Переключение задач стало еще быстрее. В телефонной книге появилась приоритетность и возможность искать контакты по карте. Номер теперь определяется не только используя контакты, но и с помощью интернета, тем самым определяя какая организация звонит. Появилось новое приложение Hangouts для общения. Печать возможно теперь совершать с помощью облачных принтеров, которые поддерживают печать через мобильные приложения. Файлы моментально сохраняются на диск

Google. Веб приложения запускаются через Google Chrome. Сервис "Удаленное управление Android". Появились приложения для e-mail с папками и новым типом навигацией. Местоположение определяется более точно и с малым расходом заряда. Запуск приложений осуществляется в Security-Enhanced Linux. Теперь лишние данные при воспроизведении видео или включенном приложении скрываются.

- Android 5.0 Lollipop (25 июня 2014).

Google представил пятую версию операционной системы Android под названием Android Lollipop. Ключевым нововведением стала дизайнерская разработка под названием Material Design, которая полностью преобразила интерфейс операционной системы за счет добавления реалистичных световых эффектов и новых цветов и направлена на обеспечение интуитивного управления голосом и прикосновениями, на которые система будет реагировать как на настоящие материалы, например, бумагу или чернила. Вместе с Android L появится возможность использовать разные слои и измерения. Это приведет операционную систему к абсолютно новому виду: более важные приложения всегда будут выделяться на первом плане, в то время как что-то второстепенное опустится куда-то на самое дно. Еще одной очень интересной и важной особенностью стало умение нового интерфейса впитывать цвета картинок и изображений с переднего плана экрана и создавать на основе этого наиболее подходящее оформление. Material Design будет объединять все ваши устройства от смартфона до автомобиля одинаковым интерфейсом и синхронизировать их. Таким образом, Вы сможете начать просматривать ролик из Интернета на смартфоне, а закончить просмотр уже на планшете. В Android Lollipop реализован совершенно новый способ уведомлений. Теперь систему можно настроить таким образом, что все уведомления будут приходить только в определенное время, которое Вы сами для себя выбираете. Кроме того, все уведомления могут быть отобраны по приоритетам, основанным на том, от кого они, из какого приложения и в какое время происходят. Появилась возможность просматривать все уведомления в одном месте,

нажав на верхнюю часть экрана. По традиции усовершенствованная версия позволяет повысить время автономной работы устройства за счет нового энергосберегающего режима и увеличить производительность некоторых процессов в четыре раза. Также пользователю теперь доступна информация о времени, через которое аккумулятор полностью разрядится. Android Lollipop обладает доработанной функцией восстановления системы, которая полноценно вернет данные при утере устройства или его замене. Для дополнительной безопасности теперь доступна функция Smart Lock, благодаря которой разблокировка устройства становится возможной только с помощью дополнительных "привязанных" аппаратов (часы, планшет, автомобиль). Кроме этого новая операционная система обеспечивает автоматическое шифрование данных. В новой версии ОС по умолчанию реализован удаленный доступ к данным. Теперь если Вы забыли смартфон дома, есть возможность войти со своего аккаунта на другом устройстве с ОС Android 5.0.

В Lollipop 5.1, которая была выпущена 9 марта 2015 года, были проведены важные обновления. Во-первых, был разработан интуитивно понятный интерфейс для работы с Wi-fi и Bluetooth. Во-вторых, что не может не радовать - теперь система поддерживает работу с несколькими SIM-картами одновременно. Раньше это реализовывали производители своими силами, поэтому не исключались проблемы с функционалом. Теперь же система позволяет воспользоваться многими удобствами, связанными в "двухкарточностью": например, выделить каждую карту своим цветом для быстрого распознавания, какая из них сейчас используется. Android 5.1 также начал поддерживать HD Voice - конечно, если оператор сотовой связи и сам смартфон также имеет эту функцию. Ну и одно из самых главных обновлений - практически полная блокировка аппарата в случае кражи или потери при помощи системы Device Protection. Смартфон или планшет станут абсолютно бесполезны и нефункциональны, пока их настоящий владелец не пройдёт проверки по аккаунту в системе Google и не снимет блок.

- Android 6.0 Muffin (28 мая 2015 года).

Android Muffin (M) был представлен на конференции Google I/O. Несмотря на то, что видимых отличий от "Леденца" не наблюдается, обновления присутствуют в достаточном количестве. Так, теперь система включает в себя блокировку устройства по отпечаткам пальцев - ранее владельцам Android это было недоступно. Также обновлений безопасности касается и то, что в новой версии запрашиваются разрешения пользователя на доступ к тем или иным функциям аппарата - теперь ни одна операция не сможет пройти мимо обладателя смартфона. Новая технология Doze поможет сохранить аккумулятор в рабочем состоянии долгое время: она уменьшает потребление энергии фоновыми приложениями [9].

В настоящее время под управлением операционной системы Android работает большое количество разнообразных устройств. Можно указать следующих производителей: Acer, Amazon, Alcatel One Touch, Asus, Archos, Beholder, Cherrypal, Creative, Dell, Effire, EKEN, Enspert, Explay, Fly, Gigabyte, Google, Hero, Highscreen, HTC, Huawei, Kogan Technologies, Kyocera, LG, Maylong, Motorola, Nikon, Notion Ink, PocketBook, Point of View, Prestige, Philips, Qumo, Roverpad, SmartQ Devices, Samsung, Sciphone, Sunno, Sony, Sony Ericsson, Toshiba, TechFaith, Viewsonic, Vobis Highscreen, ZTE, Билайн, МегаФон, МТС и др.

В таблице 1 представлена информация о версиях Android по статистике компании Google на конец 2015 года. В таблице приведены версии Android, даты их выхода, уровень API и в процентах выражено количество устройств, которые используют различные версии этой операционной системы.

Таблица 1

Информация о версиях Android

Версия Android	Название версии Android	Дата выпуска	Уровень API	Процент устройств с Android
6.0	Marshmallow	5 октября 2015	23	0,3%
5.1	Lollipop	9 марта 2015	22	10,1%
5.0	Lollipop	3 ноября 2014	21	15,5%
4.4	KitKat	31 октября 2013	19	37,8%
4.3	Jelly Bean	24 июля 2013	18	4,1%
4.2	Jelly Bean	13 ноября 2012	17	13,9%
4.1	Jelly Bean	9 июля 2012	16	11,0%
4.0	Ice Cream Sandwich	16 декабря 2011	15	3,3%
2.3	Gingerbread	9 февраля 2012	10	3,8%
2.2	Froyo	20 мая 2010	8	0,1%

К устройствам, работающим на Android, относятся смартфоны, интернет-планшеты, электронные книги, цифровые проигрыватели, наручные часы, игровые приставки, нетбуки, смартбуки, очки Google, телевизоры и других устройства. В настоящее время ведутся разработки, и в будущем планируется поддержка автомобилей и бытовых роботов.

Достоинства операционной системы Android:

- *Большое количество приложений* — по состоянию на конец 2015 года их количество достигло 1,43 миллиона единиц.
- *Открытость мобильной платформы*. Любой пользователь может внести изменения в исходный код Android, без труда написать собственную игру или программу.
- *Многозадачность*. На данной мобильной ОС можно одновременно запускать несколько приложений — например, слушать музыку и читать книгу.
- *Поддержка от компании Google*. Интернет-гигант выпускает регулярные обновления для данной платформы. Помимо этого, существует множество альтернативных прошивок от независимых программистов.

Недостатки операционной системы Android:

- Наряду с хорошей начинкой и дисплеем, большинство смартфонов на Android приходится достаточно часто подзаряжать. Связано это с тем, что операционная система Android весьма требовательна к системным ресурсам и большое количество заряда устройства по сравнению с других операционными системами.
- Большое количество версий, приложения для которых порою несовместимы друг с другом. Определенные программы могут работать только с определенными версиями операционной системы, соответственно, если телефон морально устарел, некоторые приложения на нем могут и не запуститься.
- Большое число настроек.

В настоящее время операционная система Android является лидером на рынке мобильных устройств. Android считается серьезным конкурентом для операционных систем таких компаний, как Apple и Windows. По словам генерального директора Google Сундар Пичаи, число пользователей мобильной операционной системы компании достигло 1,4 миллиарда человек. В этой статистике считаются устройства, на которых были активированы Google сервисы. За рамками официальной статистики остаётся масса пользователей в Китае, где сервисы компании запрещены, и местные производители мобильной электроники, такие как Xiaomi и Meizu, используют свои собственные экосистемы. Операционная система Android все время развивается, обновляется и стремится к новым высотам.

1.2 Архитектура операционной системы Android и Android приложений

1.2.1 Архитектура операционной системы Android

Первое, что нужно сказать про Android, это то, что она является вариантом операционной системы Linux. Внутри системы, на самом базовом ее уровне, находится модифицированное ядро Linux, которое, как в больших

системах, нужно часто обновлять. Намного чаще, чем в других мобильных ОС, иначе вы лишитесь возможности использовать новые фишки, датчики, приложения и прочие полезные штуки. Ядро работает с приложениями и драйверами через API системы, и только на этом уровне драйверов и системных приложений можно писать программы на языке C++. Среднюю часть машины занимает платформа, которую можно сравнить с .NET от Майкрософт. Она управляется языком Java. Каждое приложение пользователя порождает процесс, когда платформа формирует для него виртуальную среду исполнения. В этот момент Java-приложения в виде кода классов и библиотек автоматически переводятся в универсальный байт-код, исполняемый на байт-машине Dalvik. Эта машина имеет много общего с Java-машиной JVM. На рисунке 2 приведена архитектура операционной системы Android с ее уровнями.

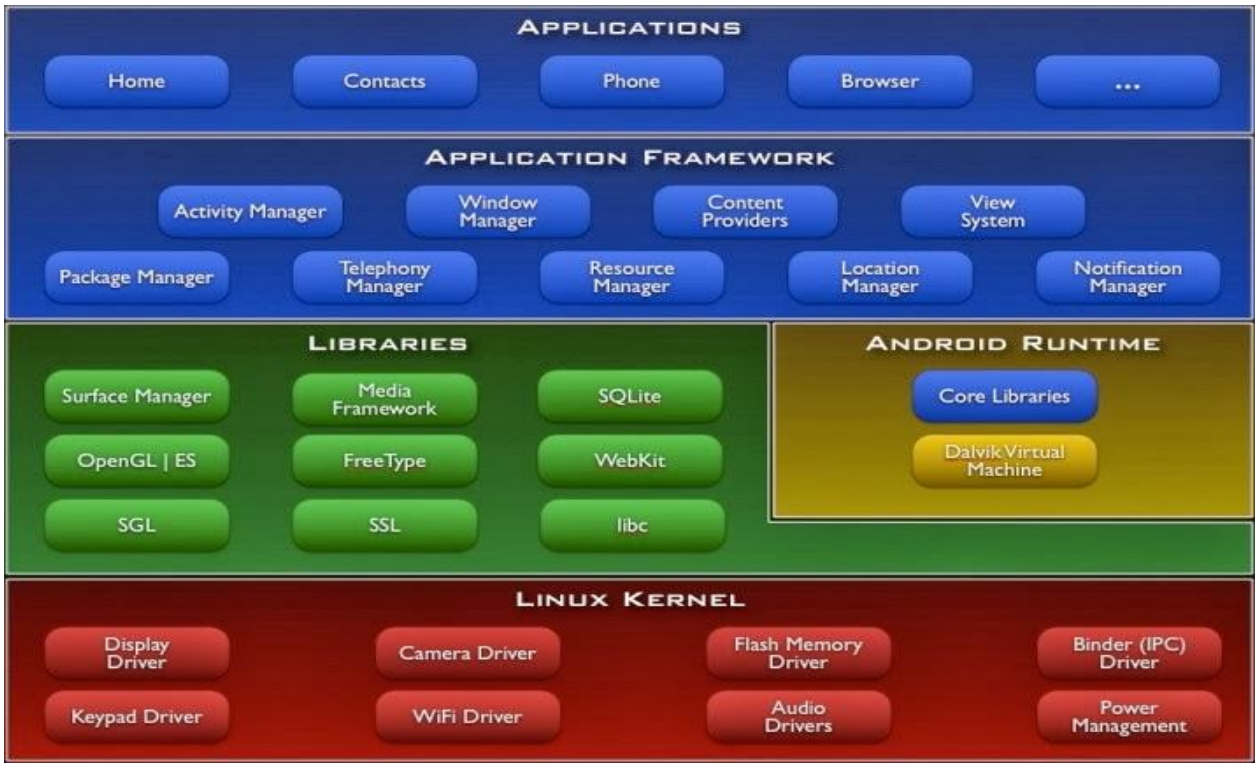

Рис. 2. Архитектура ОС Android

1) Уровень ядра Linux (Linux Kernel)

Android основан на ОС Linux версии 2.6, тем самым платформе доступны системные службы ядра, такие как: управление памятью и процесса-

ми, обеспечение безопасности, работа с сетью и драйверами. Также ядро служит слоем абстракции между аппаратным и программным обеспечением.

2) LIBRARIES (БИБЛИОТЕКИ)

На втором уровне, как программное обеспечение промежуточного слоя, лежит набор библиотек (Libraries), предназначенный для обеспечения важнейшего базового функционала для приложений. То есть именно этот уровень отвечает за предоставление реализованных алгоритмов для вышележащих уровней, поддержку файловых форматов, осуществление кодирования и декодирования информации (в пример можно привести мультимедийные кодеки), отрисовку графики и многое другое. Библиотеки реализованы на C/C++ и скомпилированы под конкретное аппаратное обеспечение устройства, вместе с которым они и поставляются производителем в предустановленном виде.

Краткое описание некоторых из них:

- Surface Manager – в ОС Android используется композитный менеджер окон, наподобие Compiz (Linux), но более упрощенный. Вместо того чтобы производить отрисовку графики напрямую в буфер дисплея, система посылает поступающие команды отрисовки в закадровый буфер, где они накапливаются вместе с другими, составляя некую композицию, а потом выводятся пользователю на экран. Это позволяет системе создавать интересные бесшовные эффекты, прозрачность окон и плавные переходы.
- Media Framework – библиотеки, реализованные на базе PacketVideo OpenCORE. С их помощью система может осуществлять запись и воспроизведение аудио и видео контента, а также вывод статических изображений. Поддерживаются многие популярные форматы, включая MPEG4, H.264, MP3, AAC, AMR, JPG и PNG.
- SQLite – легковесная и производительная реляционная СУБД, используемая в Android в качестве основного движка для работы с базами данных, используемыми приложениями для хранения информации.

- OpenGL ES (OpenGL for Embedded Systems) – подмножество графического программного интерфейса OpenGL, адаптированное для работы на встраиваемых системах. 3D-библиотеки которые, используются для высоко оптимизированной отрисовки 3D-графики, при возможности используют аппаратное ускорение. Их реализации строятся на основе API OpenGL ES 1.0.
- FreeType – библиотека для работы с битовыми картами, а также для растеризации шрифтов и осуществления операций над ними. Это высококачественный движок для шрифтов и отображения текста.
- LibWebCore – библиотеки известного быстрого браузерного движка WebKit, используемого также в десктопных браузерах Google Chrome и Apple Safari.
- SGL (Skia Graphics Engine) – открытый движок для работы с 2D-графикой. Графическая библиотека является продуктом Google и часто используется в других их программах.
- SSL - библиотеки для поддержки одноименного криптографического протокола.
- Libc – стандартная библиотека языка C, а именно её BSD реализация, настроенная для работы на устройствах на базе Linux. Носит название Bionic.

3) ANDROID RUNTIME (СРЕДА ВЫПОЛНЕНИЯ АНДРОИД)

На этом же уровне располагается Android Runtime – среда выполнения. Ключевыми её составляющими являются набор библиотек ядра (Core Libraries) и виртуальная машина Dalvik. Библиотеки обеспечивают большую часть низкоуровневой функциональности, доступной библиотекам ядра языка Java.

Каждое приложение в ОС Android запускается в собственном экземпляре виртуальной машины Dalvik. Таким образом, все работающие процессы изолированы от операционной системы и друг от друга. И вообще, архитектура Android Runtime такова, что работа программ осуществляется строго в рамках окружения виртуальной машины. Благодаря этому осуществляется

защита ядра операционной системы от возможного вреда со стороны других её составляющих. Поэтому код с ошибками или вредоносное программное обеспечение не смогут испортить Android и устройство на его базе, когда сработают. Такая защитная функция, наряду с выполнением программного кода, является одной из ключевых для надстройки Android Runtime.

Dalvik полагается на ядро Linux для выполнения основных системных низкоуровневых функций, таких как, безопасность, потоки, управление процессами и памятью. Вы можете также писать приложения на C/C++, которые будут работать непосредственно на базовом уровне ОС Linux. Хотя такая возможность и существует, необходимости в этом нет никакой. Если для приложения важны присущие C/C++ скорость и эффективность работы, Android предоставляет доступ к нативной среде разработки (NDK – Native Development Kit). Она позволяет разрабатывать приложения на C/C++ с использованием библиотек libc и libm, а также обеспечивает нативный доступ к OpenGL.

Доступ к устройствам и системным службам Android осуществляется через виртуальную машину Dalvik, которая считается промежуточным слоем. Благодаря использованию Dalvik для выполнения кода программы разработчики получают в свое распоряжение уровень абстракции, который позволяет им не беспокоиться об особенностях конструкции того или иного устройства. Виртуальная машина Dalvik может выполнять программы в исполняемом формате DEX (Dalvik Executable). Данный формат оптимизирован для использования минимального объема памяти. Исполняемый файл с расширением .dex создается путем компиляции классов Java с помощью инструмента dx, входящего в состав Android SDK. При использовании IDE Eclipse и плагина ADT (Android Development Tools) компиляция классов Java в формат .dex происходит автоматически.

Как было сказано выше, инструмент dx из Android SDK компилирует приложения, написанные на Java, в исполняемый формат (dex) виртуальной машины Dalvik. Помимо непосредственно исполняемых файлов, в состав

приложения Android входят прочие вспомогательные компоненты (такие, например, как файлы с данными и файлы ресурсов). SDK упаковывает все необходимое для установки приложения в файл с расширением .apk (Android package). Весь код в одном файле .apk считается одним приложением и этот файл используется для установки данного приложения на устройствах с ОС Android.

4) APPLICATION FRAMEWORK (КАРКАС ПРИЛОЖЕНИЙ)

Уровнем выше располагается Application Framework, иногда называемый уровнем каркаса приложений. Именно через каркасы приложений разработчики получают доступ к API, предоставляемым компонентами системы, лежащими ниже уровнем. Кроме того, благодаря архитектуре фреймворка, любому приложению предоставляются уже реализованные возможности других приложений, к которым разрешено получать доступ.

В базовый набор сервисов и систем, лежащих в основе каждого приложения и являющихся частями фреймворка, входят:

- Activity Manager – менеджер Активностей, который управляет жизненными циклами приложений, сохраняет данные об истории работы с Активностями, а также предоставляет систему навигации по ним.
- Package Manager – менеджер пакетов, управляет установленными пакетами на вашем устройстве, отвечает за установку новых и удаление существующих.
- Window Manager – менеджер окон, управляет окнами, и предоставляет для приложений более высокий уровень абстракции библиотеки Surface Manager.
- Telephony Manager – менеджер телефонии, содержит API для взаимодействия с возможностями телефонии (звонки, смс и т.п.)
- Content Providers – контент-провайдеры, управляют данными, которые одни приложения открывают для других, чтобы те могли их использовать для своей работы.

- Resource Manager – менеджер ресурсов, обеспечивает доступ к ресурсам без функциональности (не несущими кода), например, к строковым данным, графике, файлам и другим.
- View System – богатый и расширяемый набор представлений (Views), который может быть использован для создания визуальных компонентов приложений, например, списков, текстовых полей, таблиц, кнопок или даже встроенного web-браузера.
- Location Manager – менеджер местоположения, позволяет приложениям периодически получать обновленные данные о текущем географическом положении устройства.
- Notification Manager – менеджер оповещений, благодаря которому все приложения могут отображать собственные уведомления для пользователя в строке состояния.

Таким образом, благодаря Application Framework, приложения в ОС Android могут получать в своё распоряжение вспомогательный функционал, благодаря чему реализуется принцип многократного использования компонентов приложений и операционной системы. Естественно, в рамках политики безопасности. Стоит отметить, просто на понятийном уровне, что фреймворк лишь выполняет код, написанный для него, в отличие от библиотек, которые исполняются сами. Ещё одно отличие заключается в том, что фреймворк содержит в себе большое количество библиотек с разной функциональностью и назначением, в то время как библиотеки объединяют в себе наборы функций, близких по логике.

5) APPLICATIONS (ПРИЛОЖЕНИЯ)

На вершине программного стека Android лежит уровень приложений (Applications). Сюда относится набор базовых приложений, который предустановлен на ОС Android. Например, в него входят браузер, почтовый клиент, программа для отправки SMS, карты, календарь, менеджер контактов и многие другие. Список интегрированных приложений может меняться в зависимости от модели устройства и версии Android. И помимо этого базового

набора к уровню приложений относятся в принципе все приложения под платформу Android, в том числе и установленные пользователем [7].

1.2.2 Структура Android приложений

Приложения для Android в своей работе использует окна (аналогично Windows), однако в данной системе вышеуказанные окна носят иное название – Activity. Как и в Windows, каждое окно имеет свой жизненный цикл и свои особенности. При создании нового окна вызывается метод onCreate(), при разработке данный метод переопределяется и в нем происходит инициализация приложения и его компонентов. Далее вызываются методы onStart() и onResume(). Оба метода вызываются перед отображением окна при его создании, либо восстановлении (при переключении из другого приложения, при разворачивании свернутого приложения и т.п.). При сворачивании вызываются методы onPause() и onStop(). При закрытии приложения и окна вызывается onDestory(), в данном методе можно сохранить пользовательские данные и параметры. Полное описание и последовательность вызова методов можно найти на официальном сайте. Общая схема жизненного цикла приложения для Android представлена на рисунке 3.

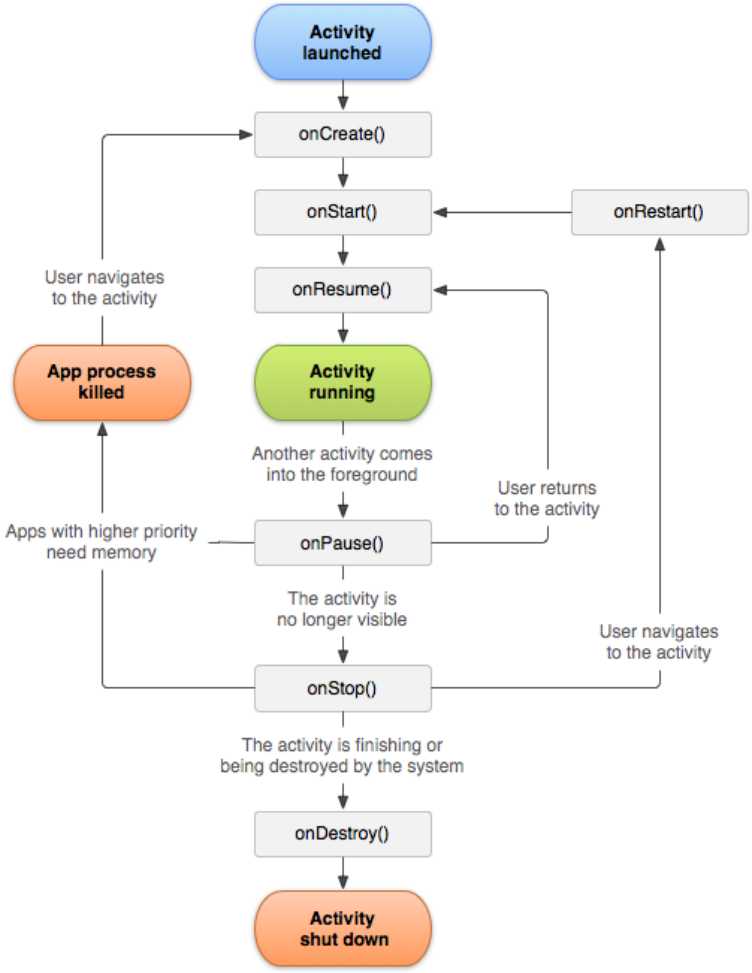

Рис. 3. Общая архитектура Android приложений

В общем случае, Android-приложение состоит из:

1. Java-классов, являющихся подклассами основных классов из Android SDK (View, Activity, ContentProvider, Service, BroadcastReciever, Intent) и Java-классов, у которых нет родителей в Android SDK.

2. Манифеста приложения.

3. Ресурсов наподобие строк, изображений и т.п.

4. Файлов.

1) Java классы:

На рисунке 4 представлена иерархия основных классов из Android SDK, с которыми работает разработчик.

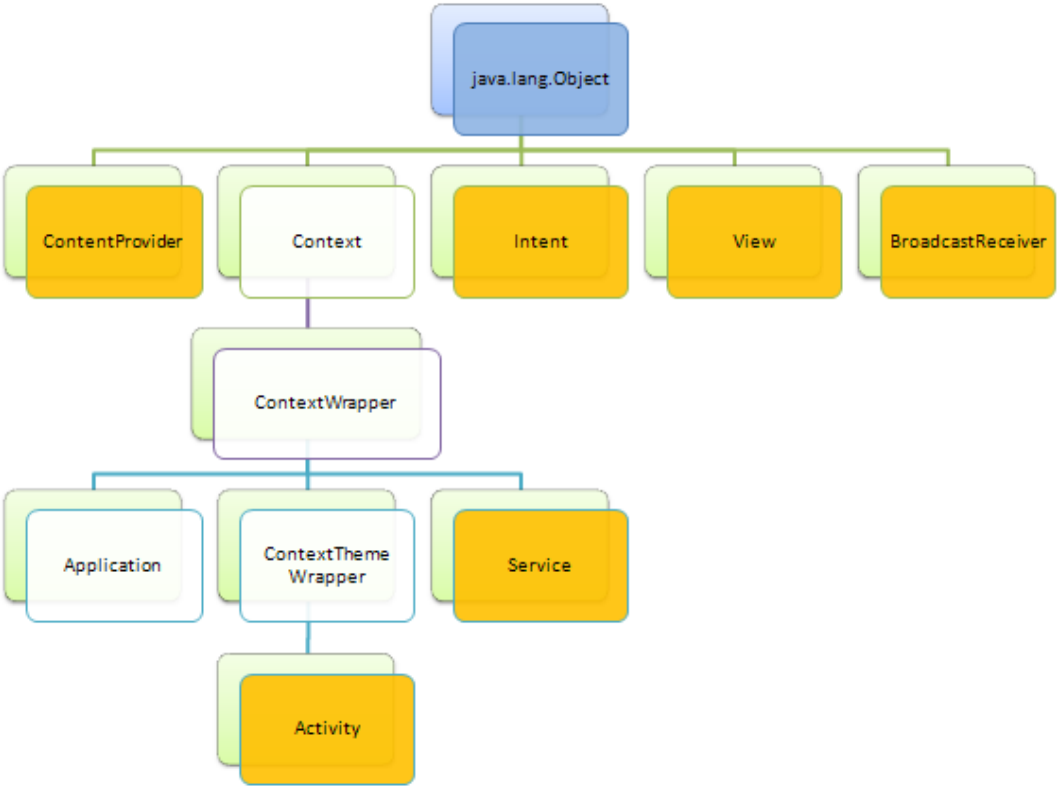

Рис. 4. Иерархия классов Android SDK используемых разработчиком

На самом деле классов намного больше, но это основные. Выделенные жёлтым — те, с которыми разработчик работает непосредственно (в частности, наследуются от них). Остальные так же важны, но они реже используются напрямую.

View — базовый класс для всех виджетов пользовательского интерфейса (GUI widgets). Интерфейс Android-приложения представляет собой дерево экземпляров наследников этого класса. Можно создать это дерево программно, но это неправильно. Пользовательский интерфейс определяется с помощью XML (файлы слоёв, layout files), а во время исполнения автоматически превращается (inflate, термин Android) в дерево соответствующих объектов.

Класс ***Activity*** и его подклассы содержат логику, лежащую за пользовательским интерфейсом. При ближайшем рассмотрении этот класс соответствует ViewModel в архитектурном шаблоне Model-View-ViewModel (MVVM). Отношение между подклассом Activity и пользовательским интерфейсом — это отношение один к одному; обычно каждый подкласс Activity

имеет только один связанный с ним слой пользовательского интерфейса, и наоборот. Activity имеет жизненный цикл.

В течении жизненного цикла Activity может находиться в одном из трёх состояний:

– Активно и выполняется — этот пользовательский интерфейс находится на переднем плане (говоря технически — на вершине стека активити).

– Приостановлено — если данный интерфейс пользователя потерял фокус, но всё ещё видим. В таком состоянии никакой код не выполняется.

– Завершено — если интерфейс пользователя невидим. В таком состоянии код не выполняется.

Код активити выполняется только когда соответствующий интерфейс пользователя видим и имеет фокус. Нет гарантии, что объект Activity и связанные с ним объекты находятся в памяти, когда активити приостановлено или завершено (очень важно помнить об этом; ранее мы уже обсуждали этот момент управления памятью в Android).

Класс ***ContentProvider*** и его подклассы представляют model в архитектуре MVVM. В большинстве практических случаев это обёртка над базой данных SQLite с немного причудливым способом доступа на основе URI. Теоретически, никто не мешает разработчику создать ContentProvider на основе чего-то ещё, кроме базы данных. Тем не менее, существующий метод query() контент-провайдера возвращает объект Cursor, который крайне похож на JDBC ResultSet интерфейсом и тем, как он работает. Поэтому вряд ли кто-то усомнится, что настоящее назначение контент-провайдеров — инкапсулировать базу данных.

Класс ***Service*** и его подклассы трудно классифицировать. Service могут выполняться даже когда процесс, в котором они работают, не на переднем плане. Так, если вы разрабатывается активити, выполняющее растянутую во времени операцию, которая должна завершиться даже работая в фоне, нужно создать Service, реализующий эту операцию, и запустить его из Activity. Service так же имеет жизненный цикл. Это означает, что он может быть уста-

новлен и запущен Android-приложением по некому условию. В большинстве случаев сервисы используются для связи с внешними серверами.

Класс ***BroadcastReceiver*** и его подклассы представляют собой «подписчика» в механизме взаимодействия издатель/подписчик, реализованном в архитектуре Android.

Конечно, разработчик под Android не ограничен одним только расширением классов из Android SDK. Он может писать собственные классы так, как захочет. Но все они будут только хелперами («helper classes») для классов из Andoird SDK.

2) Манифест Android:

Манифест Android — ещё одна важная часть Android-приложения. Идея была навеяна манифестами плагинов для Eclipse. Манифест Android представляет собой XML файл и выполняет несколько функций:

- определяет имя Java-пакета приложения. Имя пакета представляет собой уникальный идентификатор для приложения;
- описывает компоненты приложения — активити, сервисы, броадкаст-ресиверы и контент-провайдеры. Определяет имена классов, реализующие каждый из компонентов и оглашает их возможности (например, какие Intent-сообщения они могут обрабатывать). Эти объявления позволяют системе Android знать, какие компоненты и при каких условиях могут быть запущены;
- предопределяет процессы, которые будут содержать компоненты приложения;
- объявляет разрешения, которые приложение должно иметь для доступа к защищённым частям API и взаимодействия с другими приложениями;
- также объявляет разрешения, которые требуются для доступа к компонентам приложения;
- перечисляет классы Instrumentation, которые предоставляют профайлинг и другую информацию во время работы приложения. Эти объявления

присутствуют в манифесте только пока приложение разрабатывается и тестируется; они удаляются перед публикацией приложения;

- объявляет минимальный уровень Android API, который требует приложение;
- перечисляет библиотеки, с которыми приложение должно быть связано.

3) Ресурсы:

Каждое современное GUI приложение в той или иной форме использует ресурсы. Android-приложения — не исключение. Они используют следующие типы ресурсов:

- изображения;
- слои GUI (XML файлы);
- объявления меню (XML файлы);
- текстовые строки.

Способ, которым ресурсы связываются с Android-приложением — это что-то необычное. Как правило, в Java ресурсы идентифицируются строками. Такие строки могут содержать, например, путь и имя файла, содержащего изображение, или ID данной строки и т.п. Проблема в том, что ошибки в таких ссылках не могут быть обнаружены в процессе трансляции кода.

Целесообразно рассмотреть следующий пример. Файл с именем mybutton.png содержит картинку для кнопки. Разработчик совершает ошибку и набирает mybuton.png, ссылаясь на ресурс из кода. Как итог, код пытается использовать несуществующий ресурс, но компиляция пройдёт успешно. Ошибка может быть обнаружена только в ходе тестирования (а может и не быть обнаружена вовсе).

Программисты из Google нашли удобное решение этой проблемы. При сборке Android-приложения генерируется специальный Java-класс с именем R (всего лишь одна буква). Этот класс содержит несколько static final наборов данных. Каждый такой набор данных — ссылка на отдельный ресурс.

Эти ссылки используются в коде приложения для связи с ресурсами. Теперь каждая ошибка в ссылке на ресурсы проявляет себя в процессе компиляции.

4) Файлы:

Android-приложение использует несколько разных типов файлов:

- файлы «общего назначения»;
- файлы БД;
- файлы Opaque binary blob (obb) (они представляют собой зашифрованную файловую систему, которая может быть монтирована для приложения);
- закешированные файлы.

Хотя, в конечно итоге, все они только файлы Linux, имеет смысл рассматривать их как отдельные типы файлов только в разрезе обработки их различным API и отдельного хранения. Также они отделены от файлов, хранящихся во внутреннем или внешнем хранилище (последнее может отсутствовать или исчезнуть/появиться в любой момент).

Приступая к разработке мобильных приложений хорошо бы иметь представление о том, какие виды приложений существуют. Дело в том, что если удается определить к какому типу относится приложение, то становится понятнее на какие моменты в процессе его разработки необходимо обращать основное внимание. Можно выделить следующие виды приложений:

- Приложения переднего плана выполняют свои функции только, когда видимы на экране, в противном же случае их выполнение приостанавливается. Такими приложениями являются, например, игры, текстовые редакторы, видеопроигрыватели. При разработке таких приложений необходимо очень внимательно изучить жизненный цикл активности, чтобы переключения в фоновый режим и обратно проходили гладко (бесшовно), т. е. при возвращении приложения на передний план было незаметно, что оно вообще куда-то пропадало. Для достижения этой гладкости необходимо следить за тем, чтобы при входе в фоновый режим приложение сохраняло свое состояние, а при выходе на передний план восстанавливало его. Еще один важный

момент, на который обязательно надо обратить внимание при разработке приложений переднего плана, удобный и интуитивно понятный интерфейс.

- Фоновые приложения после настройки не предполагают взаимодействия с пользователем, большую часть времени находятся и работают в скрытом состоянии. Примерами таких приложений могут служить, службы экранирования звонков, SMS-автоответчики. В большинстве своем фоновые приложения нацелены на отслеживание событий, порождаемых аппаратным обеспечением, системой или другими приложениями, работают незаметно. Можно создавать совершенно невидимые сервисы, но тогда они будут неуправляемыми. Минимум действий, которые необходимо позволить пользователю: санкционирование запуска сервиса, настройка, приостановка и прерывание его работы при необходимости.

- Смешанные приложения большую часть времени работают в фоновом режиме, однако допускают взаимодействие с пользователем и после настройки. Обычно взаимодействие с пользователем сводится к уведомлению о каких-либо событиях. Примерами таких приложений могут служить мультимедиа-проигрыватели, программы для обмена текстовыми сообщениями (чаты), почтовые клиенты. Возможность реагировать на пользовательский ввод и при этом не терять работоспособности в фоновом режиме является характерной особенностью смешанных приложений. Такие приложения обычно содержат как видимые активности, так и скрытые (фоновые) сервисы, и при взаимодействии с пользователем должны учитывать свое текущее состояние. Возможно потребуется обновлять графический интерфейс, если приложение находится на переднем плане, или же посылать пользователю уведомления из фонового режима, чтобы держать его в курсе происходящего. И эти особенности необходимо учитывать при разработке подобных приложений.

- Виджеты - небольшие приложения, отображаемые в виде графического объекта на рабочем столе. Примерами могут служить, приложения для отображения динамической информации, такой как заряд батареи, про-

гноз погоды, дата и время. Разумеется, сложные приложения могут содержать элементы каждого из рассмотренных видов. Планируя разработку приложения, необходимо определить способ его использования, только после этого приступать к проектированию и непосредственно разработке [7].

Глава 2. Практическая часть

2.1 Инструменты программирования под операционную систему Android

Разработка приложений для мобильных устройств – это процесс создания приложения для небольших портативных устройств, таких как смартфоны, планшеты и др. Эти приложения могут быть предустановлены на устройство в процессе производства, загружены пользователем с помощью различных платформ для распространения ПО или являться веб-приложениями, которые обрабатываются на стороне клиента (JavaScript) или сервера.

Основным языком программирования для операционной системы Android, доказательством чего является официальная документация, считается Java. Весь API к платформе предоставлен в виде Java библиотек. Из архитектуры операционной системы Android, известно, что на самом телефоне работает Java байт-код, интерпретируемый в виртуальной машине Dalvik, который и запускается на аппарате. Кроме этого, есть NDK (native development kit) — набор инструментов и библиотек, которые позволяют скомпилировать нативный код и добавить его к приложению. Соответственно, в NDK может работать все, что компилируется в нативный код, включая интерпретаторы скриптовых языков и виртуальные машины. До недавнего времени, нельзя было создать приложение полностью написанное в нативном коде, все равно присутствовала необходимость использование Java. С недавнего времени, добавив набор нативных библиотек с системными API, стало возможно написать нативную программу от начала до конца, без использования Java.

Из выше перечисленного ясно, что можно писать приложение практически на любом языке. В реальности же, в большинстве случаев, приложения пишут на Java, иногда прикрепляют переписанные коды

углублённой настройки или сторонние библиотеки на C/C++. Исключением являются игры, которые часто пишут целиком или почти целиком на C++. Примером программ для создания игр под Android могут служить: Unreal Engine 4, Unity 3D, Marmalade, Project Anarchy (Havok/Intel), GameMaker: Studio, Corona Game Edition, Cocos2Dx, AppGameKit, libgdx, AndEngine. Примерами сторонних сред программирования, с поддержкой языков помимо Java, являются: Delphi XE5 с языком программирования Pascal и Xamarin с C#. Примером сред программирования с языком Java можно привести Eclipse, Android Studio, а так же NetBeans, IntelliJ IDEA. Две последние IDE работают при помощи плагина для Kenai для Android. Если подходить к вопросу программирования на Android OS не используя кроссплатформенность, то выделяются две основные IDE Android Studio и Eclipse. До появления Android Studio, Eclipse была самой распространенной средой разработки. Большая часть уроков по программированию написана с использованием IDE Eclipse. В настоящее время основной средой разработки считается Android Studio.

2.1.1 Основные среды программирования

Eclipse — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается Eclipse Foundation.

Eclipse служит в первую очередь платформой для разработки расширений, чем он и завоевал популярность: любой разработчик может расширить Eclipse своими модулями. Уже существуют Java Development Tools (JDT), C/C++ Development Tools (CDT), разрабатываемые инженерами QNX совместно с IBM, и средства для языков Ada (GNATbench, Hibachi), COBOL, FORTRAN, PHP, X10 (X10DT) и прочие от различных разработчиков. Множество расширений дополняет среду Eclipse диспетчерами для работы с базами данных, серверами приложений и другое [25].

Eclipse JDT (Java Development Tools) — наиболее известный модуль,

нацеленный на групповую разработку: среда интегрирована с системами управления версиями — CVS, GIT в основной поставке, для других систем (например, Subversion, MS SourceSafe) существуют плагины. Также предлагает поддержку связи между IDE и системой управления задачами (ошибками). В основной поставке включена поддержка трекера ошибок Bugzilla, также имеется множество расширений для поддержки других трекеров (Trac, Jira и др.). В силу бесплатности и высокого качества, Eclipse во многих организациях является корпоративным стандартом для разработки приложений.

Eclipse написана на Java, потому является платформо-независимым продуктом, за исключением библиотеки SWT, которая разрабатывается для всех распространённых платформ (см. ниже). Библиотека SWT используется вместо стандартной для Java библиотеки Swing. Она полностью опирается на нижележащую платформу (операционную систему), что обеспечивает быстроту и натуральный внешний вид пользовательского интерфейса, но иногда вызывает на разных платформах проблемы совместимости и устойчивости приложений. Для разработки под операционную систему Android в IDE Eclipse используется плагин Android Development Tools. Android Development Tools (ADT) – плагин обеспечивающий возможность использования инструментов Android SDK для разработки приложений Android с помощью интегрированной среды разработки Eclipse. На рисунке 5 представлена рабочая область программы Eclipse.

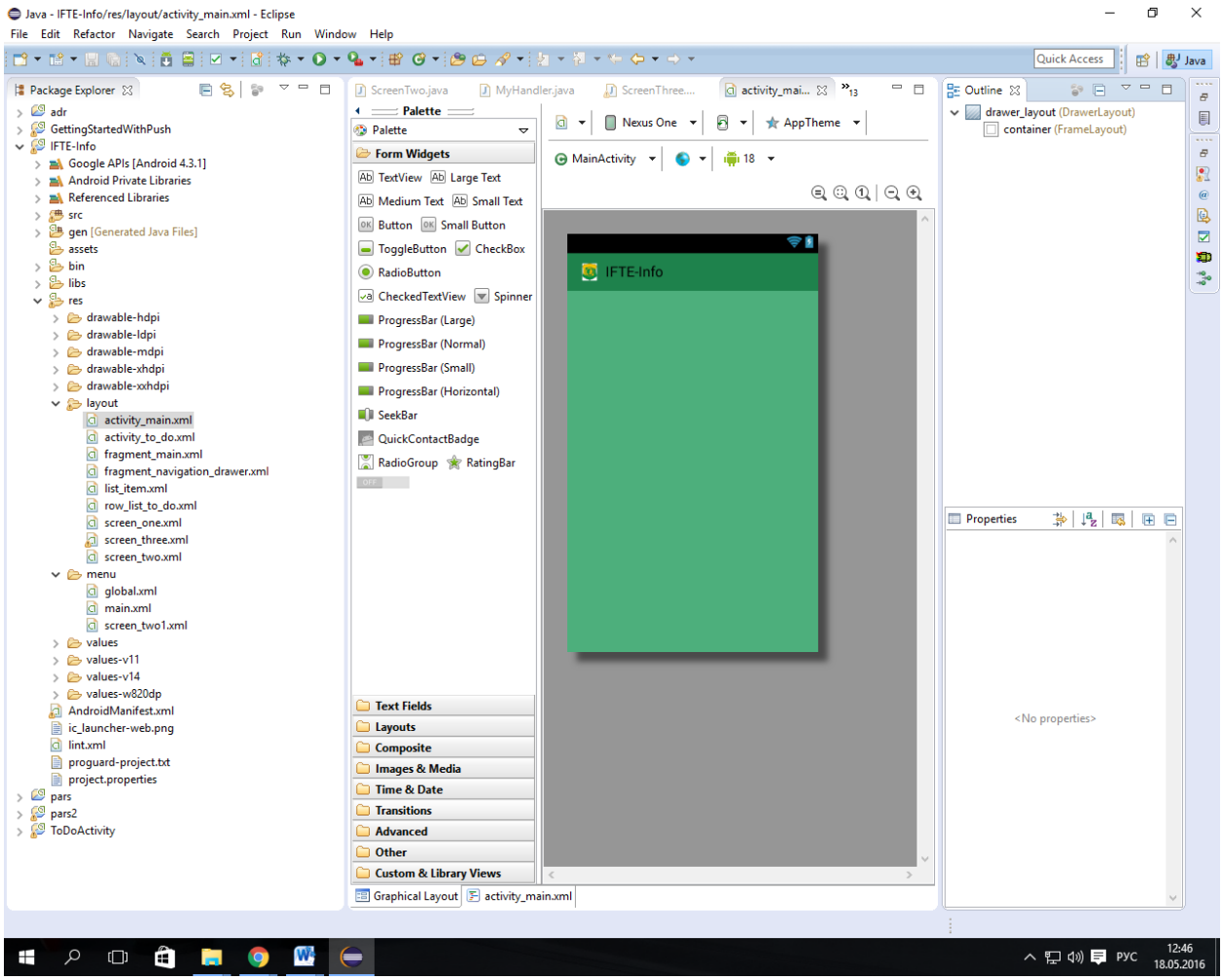

Рис. 5. Рабочая среда IDE Eclipse

Android Studio — это интегрированная среда разработки (IDE) для работы с платформой Android, анонсированная 16 мая 2013 года на конференции Google I/O.

Android Studio, основанная на программном обеспечении IntelliJ IDEA от компании JetBrains, официальное средство разработки Android приложений. Данная среда разработки доступна для Windows, OS X и Linux.

Новые функции появляются с каждой новой версией Android Studio. На данный момент доступны следующие функции:

- Расширенный редактор макетов: WYSIWYG, способность работать с UI компонентами при помощи Drag-and-Drop, функция предпросмотра макета на нескольких конфигурациях экрана.
- Сборка приложений, основанная на Gradle.

- Различные виды сборок и генерация нескольких .apk файлов.
- Рефакторинг кода.
- Статический анализатор кода (Lint), позволяющий находить проблемы производительности, несовместимости версий и другое.
- Встроенный ProGuard и утилита для подписки приложений.
- Шаблоны основных макетов и компонентов Android.
- Поддержка разработки приложений для Android Wear и Android TV.
- Встроенная поддержка Google Cloud Platform, которая включает в себя интеграцию с сервисами Google Cloud Messaging и App Engine.
- Android Studio 2.1 поддерживает Android N Preview SDK, а это значит, что разработчики смогут начать работу по созданию приложения для новой программной платформы.
- Новая версия Android Studio 2.1 способна работать с обновленным компилятором Jack, а также получила улучшенную поддержку Java 8 и усовершенствованную функцию Instant Run.

На рисунке 6 представлены логотип и рабочая область программы Android Studio.

Рис. 6. Логотип и рабочая среда Android Studio

2.1.2 Языки программирования

Основным языком программирования под Android является Java. **Java** - объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle).

Приложения Java обычно транслируется в специальный байт-код, поэтому они могут работать на любой виртуальной Java-машине (JVM) вне зависимости от компьютерной архитектуры.

Дата официального выпуска - 23 мая 1995 года.

Java является основой практически для всех типов сетевых приложений и всеобщим стандартом для разработки и распространения встроенных и мобильных приложений, игр, веб-контента и корпоративного программного обеспечения.

В мире насчитывается более 9 миллионов специалистов, разрабатывающих приложения на Java, которая позволяет эффективно разрабатывать, внедрять и использовать превосходные приложения и услуги.

От портативных компьютеров до центров сбора данных, от игровых консолей до суперкомпьютеров, используемых для научных разработок, от сотовых телефонов до сети Интернет — Java используется повсюду.

– Java используется на 97% корпоративных настольных ПК;
– Java используется на 89% настольных ПК в США;
– 9 млн разработчиков на Java в мире;
– Инструмент номер 1 среди разработчиков;
– Программа номер 1 среди разработчиков;
– Java используется в 3 млрд мобильных телефонов;
– Java входит в комплект поставки 100% всех проигрывателей дисков Blu-ray;
– Используется 5 млн Java Card;
– Java используется в 125 млн ТВ-устройств;

– 5 из 5 основных производителей оригинального оборудования включают в комплект поставки Java ME.

Основные особенности языка Java при использования Android:

Программы на Java транслируются в байт-код, выполняемый виртуальной машиной Java (JVM) - программой, обрабатывающей байтовый код и передающей инструкции оборудованию как интерпретатор.

Достоинством подобного способа выполнения программ является полная независимость байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина.

Другой важной особенностью технологии Java является гибкая система безопасности благодаря тому, что исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка несанкционированного доступа к данным или соединения с другим компьютером) вызывают немедленное прерывание.

Часто к недостаткам концепции виртуальной машины относят то, что исполнение байт-кода виртуальной машиной может снижать производительность программ и алгоритмов, реализованных на языке Java.

В последнее время был внесен ряд усовершенствований, которые несколько увеличили скорость выполнения программ на Java:

– применение технологии трансляции байт-кода в машинный код непосредственно во время работы программы (JIT-технология) с возможностью сохранения версий класса в машинном коде;

– широкое использование платформенно-ориентированного кода (native-код) в стандартных библиотеках;

– аппаратные средства, обеспечивающие ускоренную обработку байт-кода (например, технология Jazelle, поддерживаемая некоторыми процессорами фирмы ARM).

По данным сайта shootout.alioth.debian.org, для семи разных задач время выполнения на Java составляет в среднем в полтора-два раза больше, чем для C/C++, в некоторых случаях Java быстрее, а в отдельных случаях в 7 раз медленнее. С другой стороны, для большинства из них потребление памяти Java-машиной было в 10-30 раз больше, чем программой на C/C++.

Также примечательно исследование, проведённое компанией Google, согласно которому отмечается существенно более низкая производительность и большее потребление памяти в тестовых примерах на Java в сравнении с аналогичными программами на C++.

В процессе программиования под Android используется Java Development Kit (JDK) - бесплатно распространяемый компанией Oracle Corporation комплект разработчика приложений на языке Java, включающий в себя компилятор Java (javac), стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java (JRE).

С использованием языка программирования Java, Anroid приложения для работы с файлами используют язык XML. **XML** - весьма популярный формат для обмена информации, хотя постепенно сдаёт свои позиции более быстрому и экономному формату JSON. Тем не менее, XML по-прежнему используется многими веб-сервисами, которые предоставляют котировки валюты, прогноз погоды, очаги землетрясений и т.п. Разметка экрана, ресурсы и другие файлы приложения состоят из XML-файлов. Читать такие файлы человеку не очень удобно. Поэтому необходимо использовать специальные инструменты для обработки текста - парсеры, которые преобразуют XML-файл в удобно читаемый текст.

Библиотеки Android имеют несколько наборов классов для работы с XML-документами с произвольной структурой и содержанием. Поддерживаются технологии SAX (Simple API for XML), XML Pull Parser, Limited DOM Level 2 core support (объектная модель документов) и др.

2.1.3 Инструмент Android Software Development Kit

Android Software Development Kit (SDK) – инструментальный набор

включающий средства, используемые для разработки, тестирования и отладки приложений Android. Android SDK – включает в себя разнообразные библиотеки, документацию и инструменты, которые помогают разрабатывать мобильные приложения для платформы Android.

Структура Android SDK:

- API Android SDK-API библиотеки Android, предоставляемые для разработки приложений. На рисунке 7 представлен менеджер библиотек Android SDK.

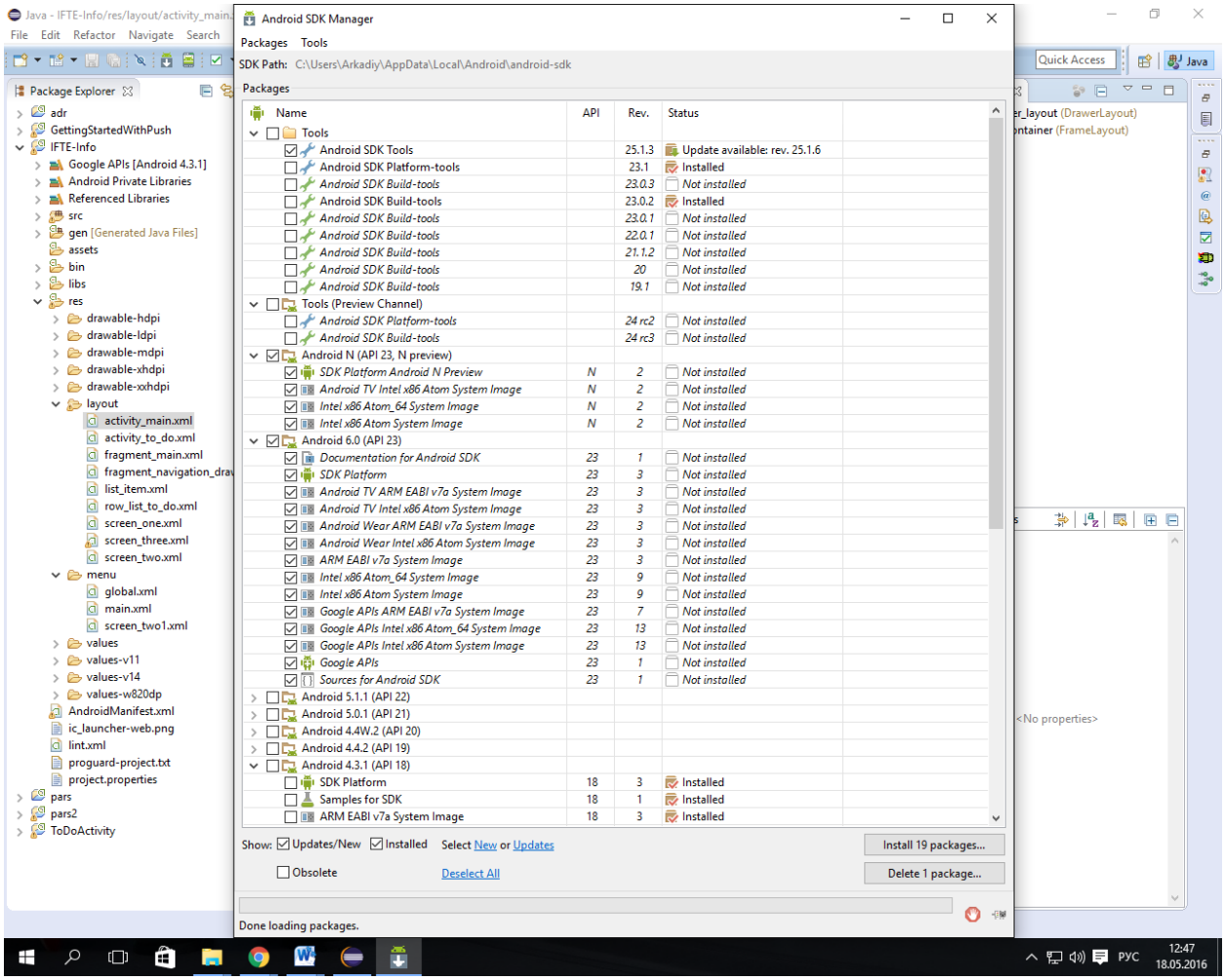

Рис. 7. Менеджер библиотек Android SDK

- Документация SDK-включает обширную справочную информацию, детализирующую, что включено в каждый пакет и класс и как это использовать при разработке приложений.

- AVD (Android Virtual Device) -интерактивный эмулятор мобильного устройства Android. Используя эмулятор, можно запускать и тестировать

приложения без использования реального Android устройства. На рисунке 8 представлена рабочая область AVD.

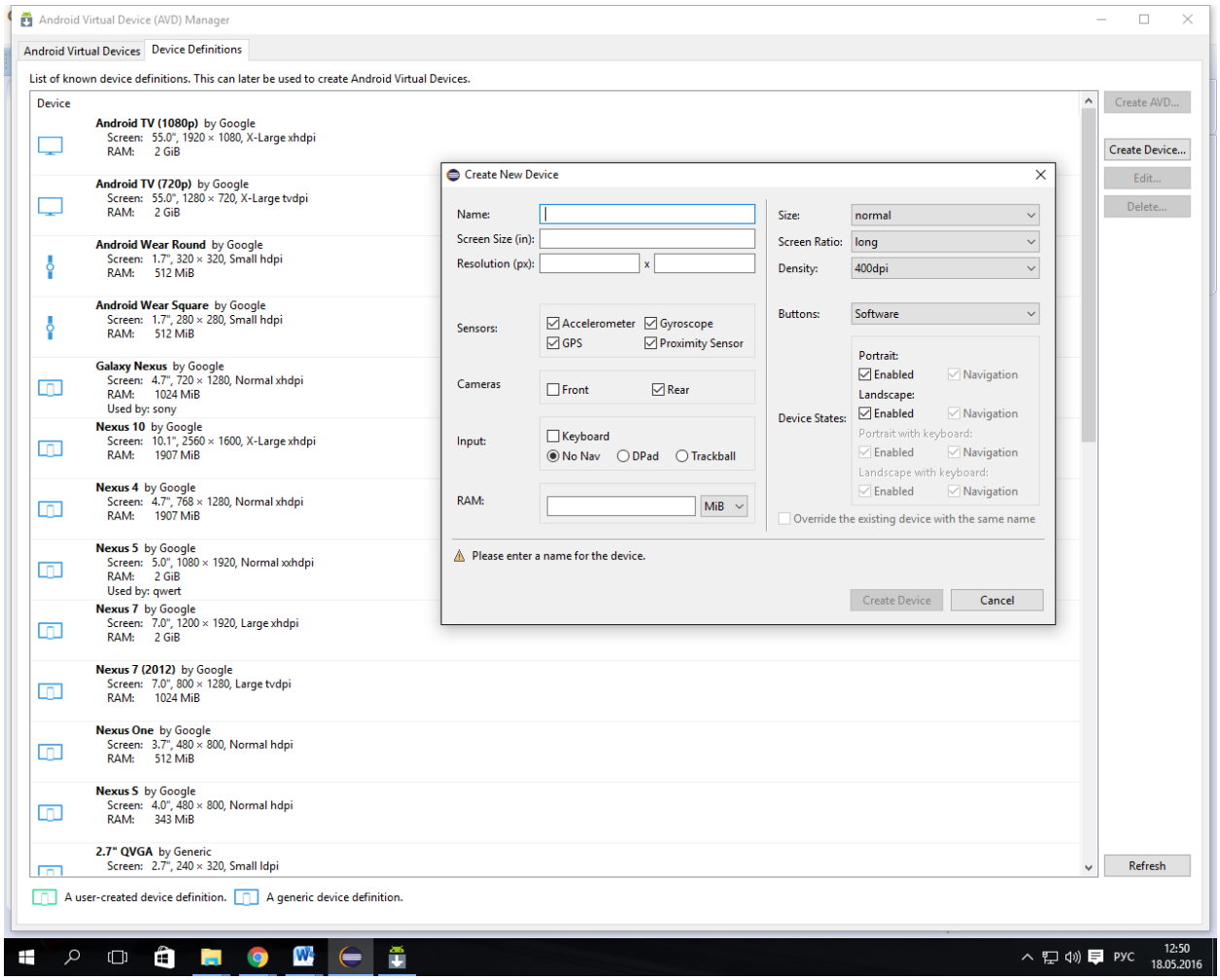

Рис. 8. Интерактивный эмулятор мобильного устройства Android

- Development Tools – SDK включает несколько инструментальных средств для разработки, которые позволяют компилировать и отлаживать создаваемые приложения.

- Sample Code – Android SDK предоставляет типовые приложения, которые демонстрируют некоторые из возможностей Android, и простые программы, которые показывают, как использовать индивидуальные особенности API в вашем коде.

Версии SDK и Android API Level

Перед началом разработки приложений для Android полезно понять общий подход платформы к управлению изменением API. Также важно понять Android API Level (Идентификатор уровня API) и его роль в

обеспечении совместимости вашего приложения с устройствами, на которых оно будет устанавливаться.

Уровень API – целочисленное значение, которое однозначно определяет версию API платформы Android. Платформа обеспечивает структуры API, которые приложения могут использовать для взаимодействия с системой Android. Каждая следующая версия платформы Android может включать обновления API.

Обновления API-структуры разработаны так, чтобы новый API оставался совместимым с более ранними версиями API. Таким образом, большинство изменений в API является совокупным и вводит новые функциональные возможности или исправляет предыдущие. Поскольку часть API постоянно обновляется, устаревшие API не рекомендуются к использованию, но не удаляются из соображений совместимости с имеющимися приложениями.

Таблица 2

Соотношение версий Android и уровня API, поддерживаемые ими

Версия платформы Android	Название версии Android	Уровень API
6.0	Marshmallow	23
5.1	Lollipop	22
5.0	Lollipop	21
4.4	KitKat	19
4.3	Jelly Bean	18
4.2	Jelly Bean	17
4.1	Jelly Bean	16
4.0	Ice Cream Sandwich	15
2.3	Gingerbread	10
2.2	Froyo	8

Уровень API, который использует приложение для Android, определяется целочисленным идентификатором, который указывается в файле конфигурации каждого Android-приложения. Таблица 2 определяет соответствие уровня API и версии платформы Android.

Инструменты для разработки и отладки приложений:

Кроме эмулятора, SDK также включает множество других инструментальных средств для отладки и установки создаваемых приложений. При разработке приложения для Android с помощью IDE Eclipse, многие инструменты командной строки, входящие в состав SDK, уже используются при сборке и компиляции проекта. Однако кроме них SDK содержит еще ряд полезных инструментов для разработки и отладки приложений:

- Android – важный инструмент разработки, запускаемый из командной строки, который позволяет создавать, удалять и конфигурировать виртуальные устройства, создавать и обновлять Android проекты (при работе вне среды Eclipse) и обновлять Android SDK новыми платформами, дополнениями и документацией;
- Dalvik Debug Monitor Service (DDMS) – интегрированный с Dalvik Virtual Machine, стандартной виртуальной машиной платформы Android, этот инструмент позволяет управлять процессам и на эмуляторе ил и устройстве, а также помогает в отладке приложений. Вы можете использовать этот сервис для завершения процессов, выбора определенного процесса для отладки, генерирования трассировочных данных, просмотра "кучи" или информации о потоках, делать скриншоты эмулятора ил и устройства и многое другое;
- Hierarchy Viewer— визуальный инструмент, который позволяет отлаживать и оптимизировать пользовательский интерфейс разрабатываемого приложения. Он показывает визуальное дерево иерархии представлений, анализирует быстродействие перерисовки графических изображений на экране и может выполнять еще много других функций для анализа графического интерфейса приложений;
- Layoutopt— инструмент командной строки, который помогает оптимизировать схемы разметки и иерархии разметок в создаваемом приложении. Необходим для решения проблем при создании сложных

графических интерфейсов, которые могут затрагивать производительность приложения;

- Draw 9-patch – графический редактор, который позволяет легко создавать NinePatch графику для графического интерфейса разрабатываемых приложений;
- sqlite3 – инструмент для доступа к файлам данных SQLite, созданных и используемых приложениями для Android;
- Traceview – этот инструмент выдает графический анализ трассировочных логов, которые можно генерировать из приложений;
- mksdcard – инструмент для создания образа диска, который вы можете использовать в эмуляторе для симуляции наличия внешней карты памяти (например, карты SD).

Наиболее важный из них— эмулятор мобильного устройства, однако в состав SDK входят и другие инструменты для отладки, упаковки и инсталляции ваших приложений на эмулятор.

2.2 Разработка приложения под операционную систему Android

2.2.1 Постановка задачи. Требования к приложению. Подготовительные действия

Задачами, выполняемыми во время прохождения преддипломной практики, является изучение интегрированной среды разработки Eclipse под операционную систему Android и написание приложения. Программирование под Android, благодаря гибкости этой платформы, позволяет создавать полезные и запоминающиеся мобильные приложения практически под любые нужды. В процессе работы было установлено и использовано следующее программное обеспечение:

- Java Development Kit (JDK)
- Eclipse IDE;
- Android Software Development Kit (SDK);

- Android Development Tools (ADT);

-Microsoft Azure;

-Microsoft SQL Server Management Studio.

Для тестирования приложения использовался телефон Sony Xperia V работающий на операционной системе Android версии 4.3 Jelly Bean и модуль Android Virtual Device инструмента Android Software Development Kit.

Создаваемое приложение называется ***IFTE-Info***.

Задача приложения: информировать пользователя о новостях Института физики, экономики и технологий УрГПУ. Приложение отчасти является аналогом сайта факультета (мобильной версией), а так же дополнительным средством информирования студентов.

В приложение заложены три основных функции:

1) предоставление пользователю расписания,
2) загрузка новостей с сайта УрГПУ,
3) вывод объявлений из базы данных хранящейся на облачном сервисе Microsoft Azure.

Каждая функция выполняется в отдельном классе фрагментов, которые вызываться при переходе по вкладкам меню с соответствующими названиями. В приложении используются следующие библиотеки:

- jsoup – для реализации парсинга новостей с сайта УРГПУ;

- mobileservices, json, guava – для подключения к базе данных на сервисе Azure и доступа к базе данных.

-notifications, google-play-services – для оправки пользователям push-уведомлений.

Исполняемый код на языке Java из документов:

-MainAcivity – главное Ativity приложения в котором прописан жизненный цикл и осуществляется переход между фрагментами;

-Класс фрагментов, включающий ScreenOne, ScreenTwo и ScreenThree, в которых производится выполнение отдельных функции приложения;

- NavigationDrawerFragment в котором прописана структура работы выдвигающегося меню с вкладками;

- MyHandler – осуществляет работу Push уведомлений на стороне пользователя;

- ToDoItem и ToDoItemAdapter, выполняющий работу с базой данных на сервисе Azure.

Для функционирования приложения в файл AndroidManifest добавлены разрешения на доступ в интернет, использование сервиса google для push-уведомлений и на доступ к базе данных.

Основная структура проекта приведена на рисунке 9.

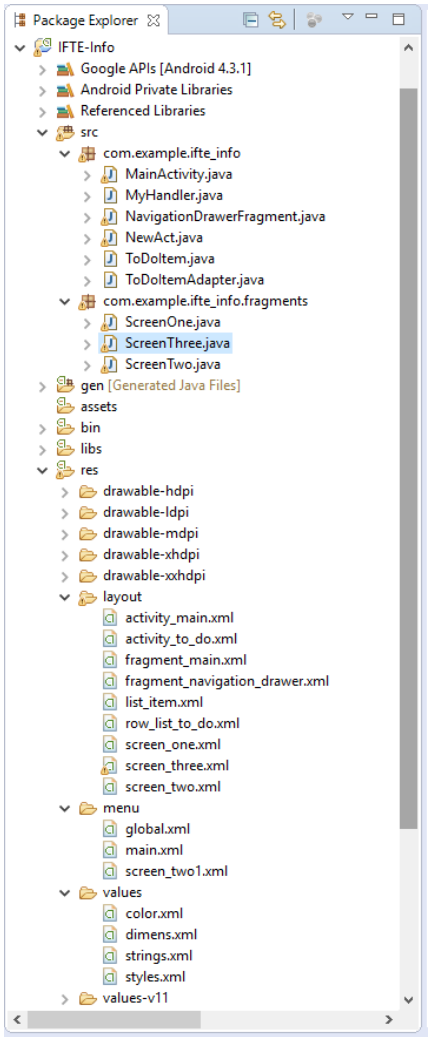

Рис. 9. Структура проекта IFTE-Info

Интерфейс Android программ создается при помощи языка разметки XML. При помощи компонентов настраивается окно программы (layout). Редактирование параметров можно выполнять в режиме визуального програм-

мирования или путем редактирования xml кода. В интерфейсе программы IFTE-Info используется шаблон Navigation drawer activity. В шаблоне присутствует Action Bar (панель в верхней части приложения) с кнопками вызова меню вкладок и настроек. В приложении IFTE-Info кнопка настроек используется для вывода на экран сообщения с информацией о приложении. На рисунке 10 представлен внешний вид приложения с открытым меню вкладок.

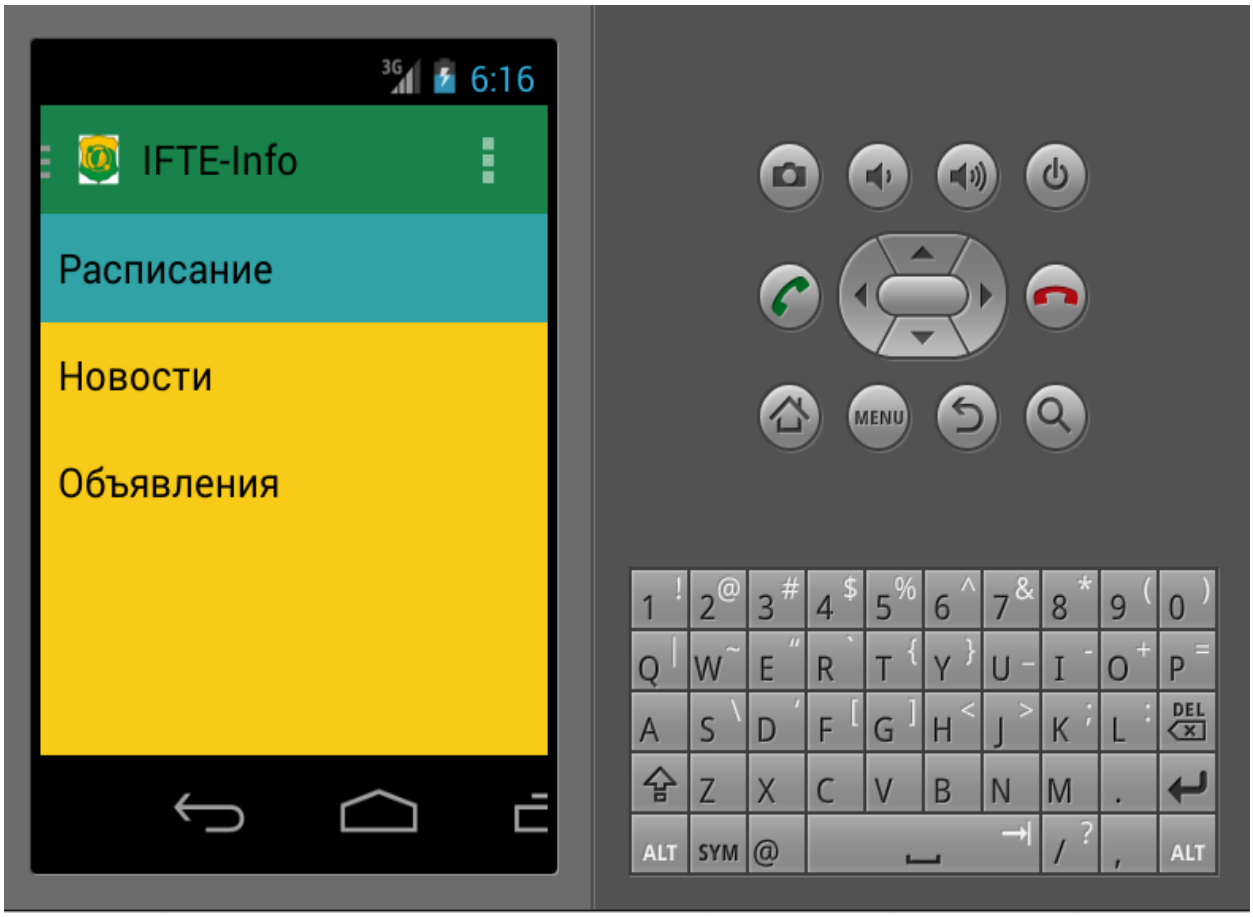

Рис. 10. Окно вкладок приложения IFTE-Info

В фоне приложения используется темно зеленый цвет, который по своим свойствам удобен для восприятия информации пользователем. Фоном меню вкладок выбран желтый цвет, который гармонирует с темно зеленым и по своим свойствам несет функцию привлечения внимания пользователя. Цвет шрифта выбран черный. В совокупности использование этих цветов делает приложение удобным для восприятия и соответствует расцветки эмблемы Института физики, экономики и технологий УРГПУ.

2.2.2 Работа элемента Расписание

Фрагмент ScreenOne.java реализует работу вкладки Расписание. В представлении он использует screen_one.xml layout. На нем расположены такие элементы как TextView, RadioGroup, Spriner и ImageView. TextView выводит текстовое сообщение для пользователя, с призывом выбрать курс и группу. RadioGroup представляет из себя набор RadioButton – кнопок с двумя положениями включено или выключено. В RadioGroup предоставляется выбор только одной кнопки из RadioButton. В приложении RadioGroup используется для выбора курса. После нажатия выбора курса на экран выводится Spriner – сворачивающийся список, в котором пользователь выбирает группу. После выбора курса и группы на основную часть экрана приложения выводиться изображение расписания помещенное в ImageView. Работы вкладки Расписание представлен на рисунке 11.

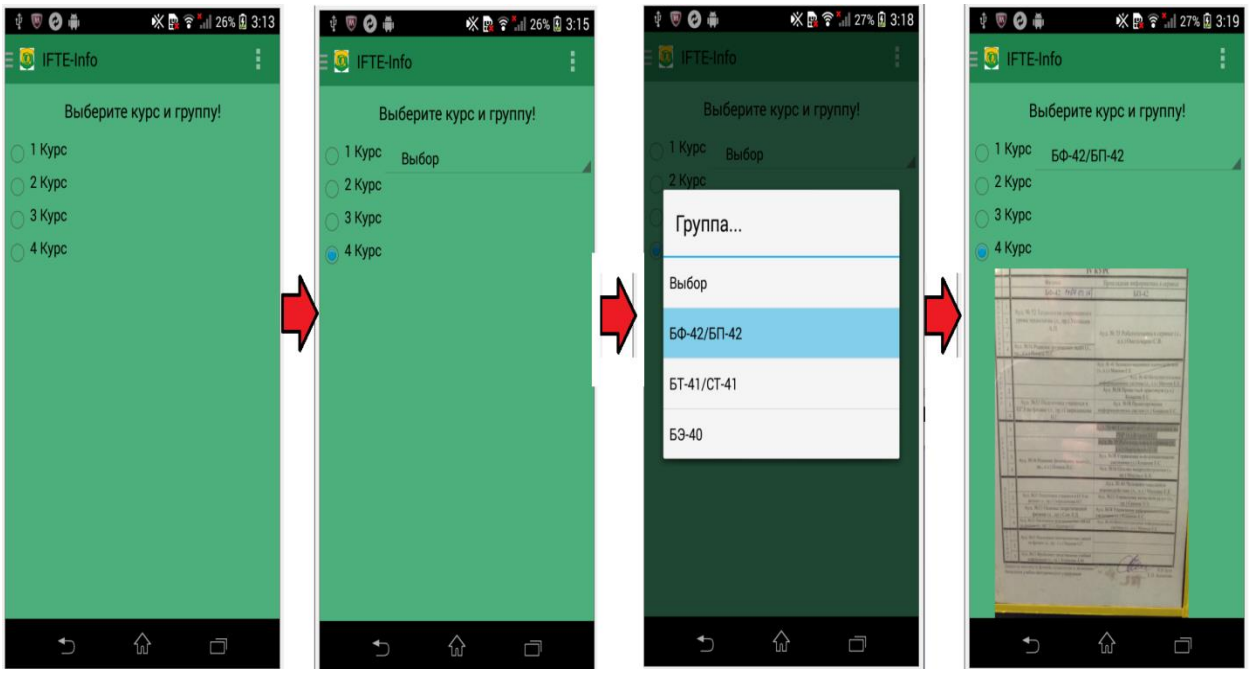

Рис. 11. Работа вкладки Расписание приложения IFTE-Info

При переключении RadioButton в Spriner загружается массив имен соответствующий каждому курсу, а при выборе группы в ImageView загружается изображение с расписанием из папки drawable. При нажатии на изображение оно увеличиваться на весть экран для просмотра. Приложении 1 приведен пример кода ScreenOne.java выполняющий переключение и вывод изображения.

2.2.3 Работа элемента Новости

Фрагмент ScreenTwo.java реализует работу вкладки Новости. Информации берется с сайта УРГПУ из раздела новости посредством парсинга. *Парсинг* – это синтаксический анализ сайтов, который автоматически производится парсером – специальной программой или скриптом. Характер парсинга определяется заданием получить определенную информацию со страниц сайта, параметры анализа заранее задаются. В приложении IFTE-Info парсинг осуществляется при помощи библиотеки Jsoup. Сначала приложение находит заголовки новостей и помещает их в элемент список ListView screen_two.xml layout. Для реализации прокручивания списка новостей по вертикали элементы ListView помещены в ScrollView. При нажатии на заголовок одной из новости на экран приложения выводится ее содержание, которае так же по средстам парсинга помещается в элемент TextView. Для реализации прокручивания содержания новости по вертикали элемент TextView помещен в ScrollView. Работа вкладки Новости продемонстрирована на рисунке 12. На рисунке 13 представлен скриншот раздела новости сайта УРГПУ для сравнения с работой вкладки Новости. Элемент кода ScreenTwo.java выполняющий парсинг представлен в приложении 2.

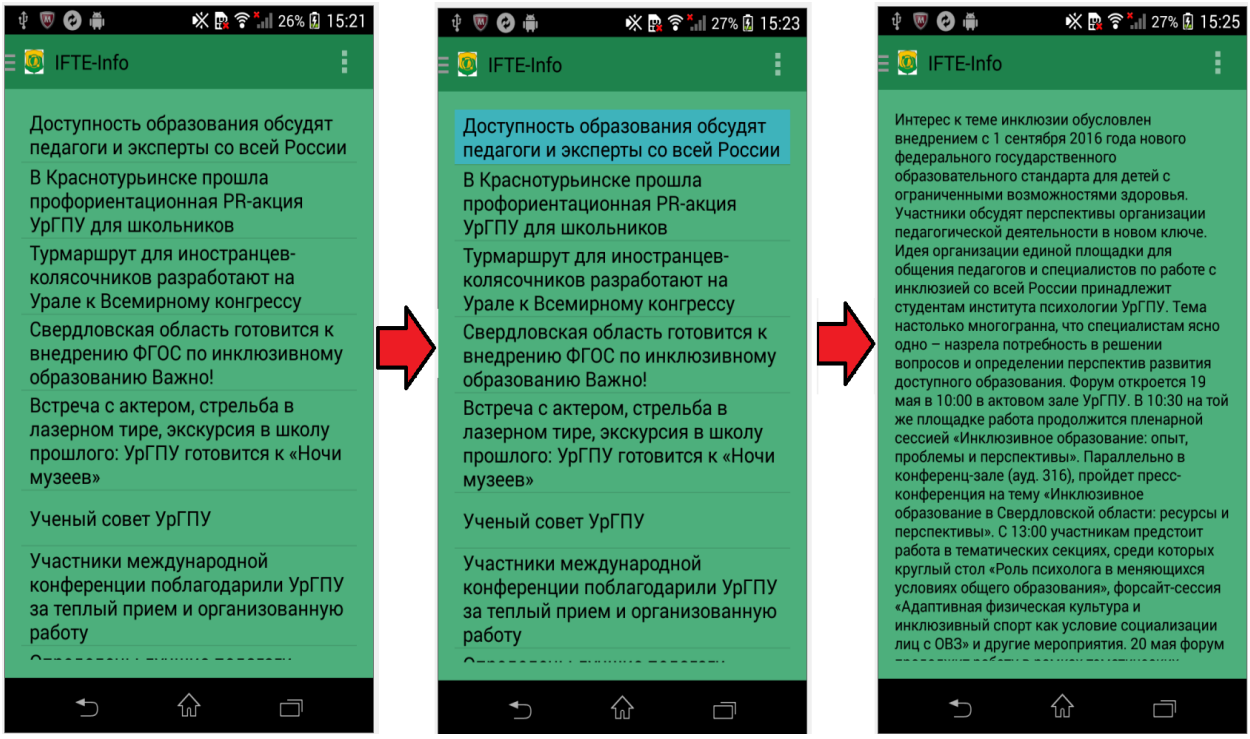

Рис. 12. Работа вкладки Новости приложения IFTE-Info

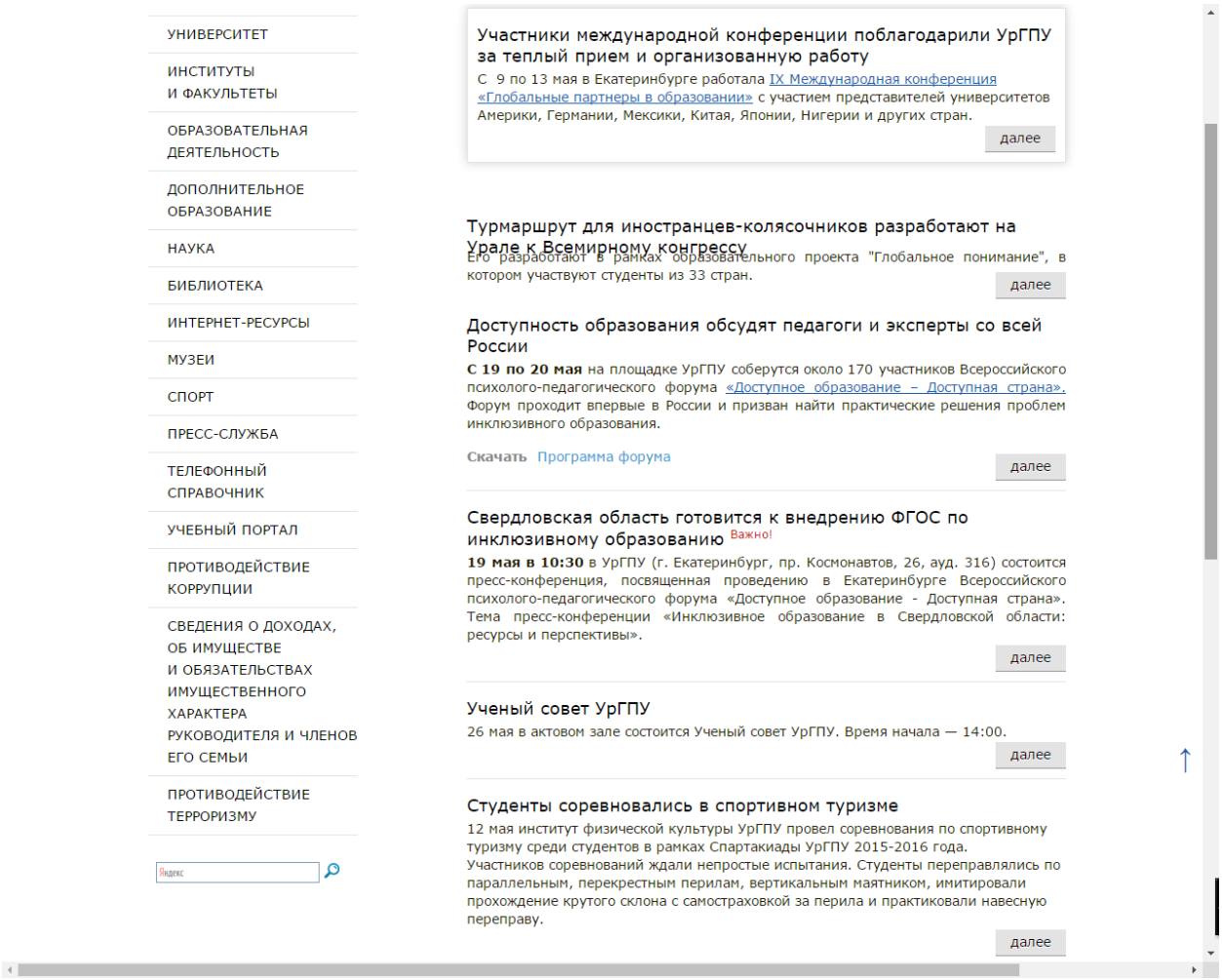

Рис. 13. Скриншот страницы новостей сайта УрГПУ

2.2.4 Работа элемента Объявления

Вкладка *Объявления* предназначена для загрузки информации из базы данных из удаленного источника. Для реализации работы вкладки были использованы возможности облачного сервиса Microsoft Azure. Microsoft Azure — это облачная платформа компании Microsoft. Предоставляет возможность разработки и выполнения приложений и хранения данных на серверах, расположенных в распределённых дата-центрах. Для разработчика Android приложений Microsoft Azure предоставляет следующие возможности:

- доступ к реляционному хранилищу с динамической схемой данных;
- интегрированная система аутентификации на базе Microsoft Account, Facebook, Google и Twitter;
- пуш-уведомления для приложений Android (и всех других платформ одновременно);
- отправка SMS и почтовых сообщений из облака через сервисы SendGrid и Twilio;
- облачное масштабирование от бесплатного уровня до высоких нагрузок.

За работу элемента Объявления отвечает файл ScreenThree.java. Для загрузки новостей из базы данных в проект были добавлены дополнительные файлы ToDoItem.java и ToDoItemAdapter.java. По аналогии с работой вкладки Новости, информация о объявлениях загружается в список ListView экрана screen_three.xml. На портале Microsoft Azure была произведена регистрация и создание проекта в сервисе мобильных приложений и создана база данных SQL. На рисунке 14 приведена рабочая область проекта на облачном сервисе.

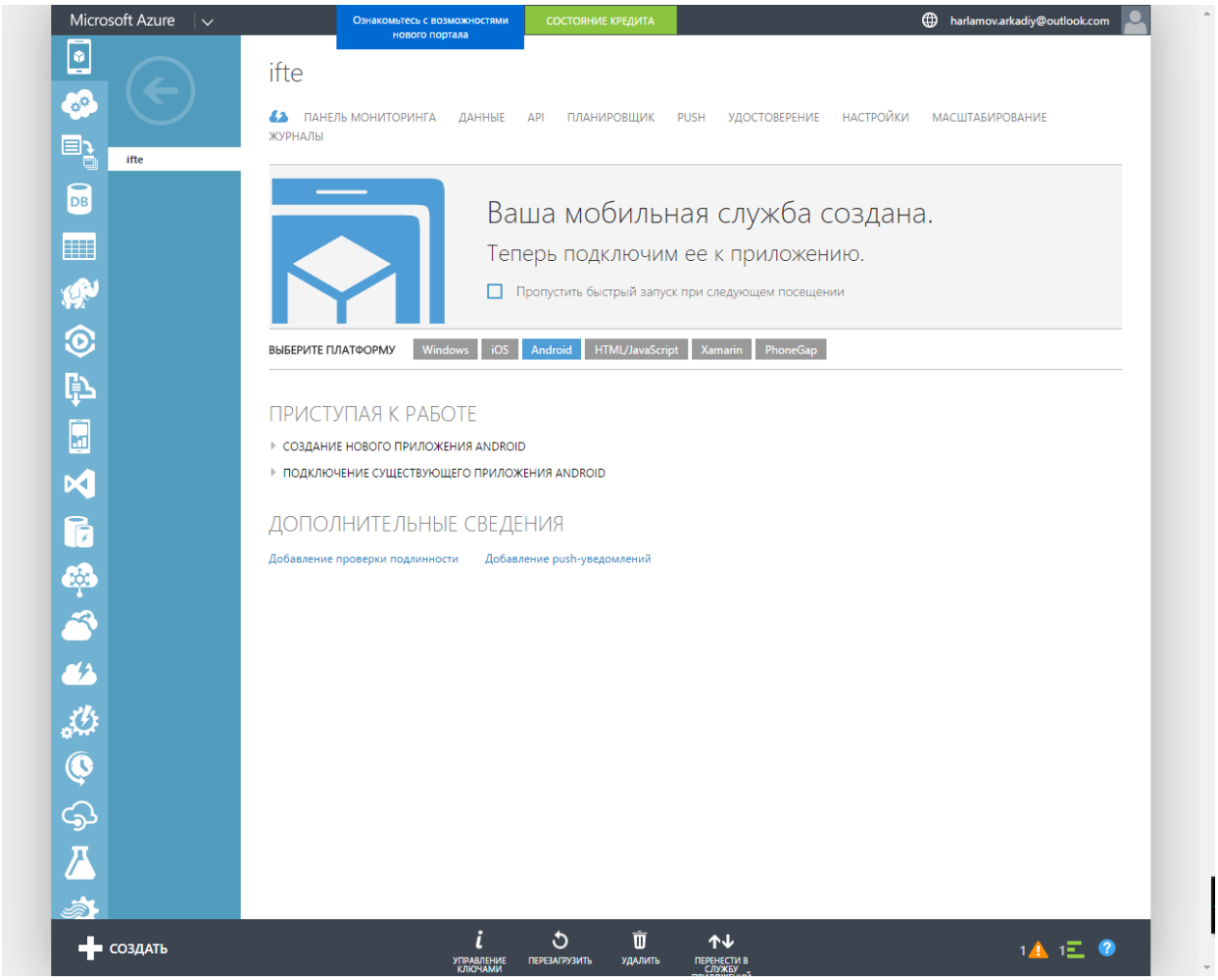

Рис. 14. Рабочая область приложения IFTE-Info на сервисе Microsoft Azure

Чтение информации из базы данных можно производить из облачного сервиса. Для внесения и изменения информации можно использовать дополнительное приложение Android или программное обеспечение Microsoft SQL Server Management Studio. SQL Server Management Studio (SSMS) — это интегрированная среда для доступа, настройки, администрирования, разработки всех компонентов SQL Server и управления ими. На рисунке 15 представлена база данных на облачном сервисе Microsoft Azure.

todoitem

ОБЗОР СКРИПТ СТОЛБЦЫ РАЗРЕШЕНИЯ

id	text	complete	__createdAt	__updatedAt	__version
3F59B8EF-8FF2-4239-B0D6...	Объявление 2	false	2016-05-19T13:50:34.078+...	2016-05-19T13:50:34.078+...	AAAAAAAAC8A=
7483D37E-88E4-4382-8B06...	Новость 2	false	2016-05-19T13:50:23.015+...	2016-05-19T13:50:23.015+...	AAAAAAAAC74=
AD813631-0F9E-4985-BD6...	Объявление	false	2016-05-19T13:50:15.453+...	2016-05-19T13:50:15.453+...	AAAAAAAAC7w=
BC7C746E-5611-40FA-BBCC...	Новость 1	false	2016-05-19T13:49:54.156+...	2016-05-19T13:49:54.344+...	AAAAAAAAC7o=

Рис. 15. БД приложения IFTE-Info в сервисе Microsoft Azure

При помощи сервиса Microsoft Azure в приложение добавлена возможность принятия push-уведомлений. Push-уведомления – это краткие всплывающие уведомления, которые появляются на экране мобильного телефона и сообщают о важных событиях и обновлениях. В IFTE-Info уведомления используются для работы вкладки Объявления. Когда необходимо проинформировать пользователя о каком-либо важном объявления, информация отправляется посредствам push-уведомлений. Даже если пользователь не использует приложение, он будет осведомлен об объявлении добавленном в базу данных. Работа вкладки Объявления приведена на рисунке 16.

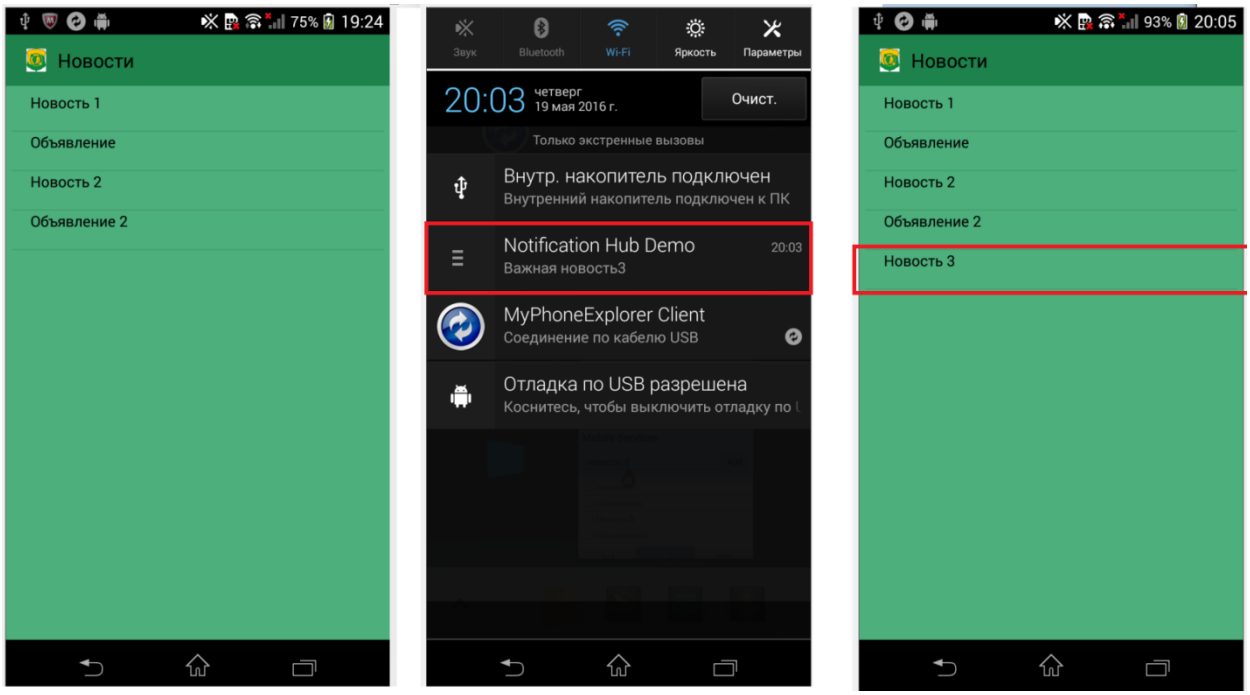

Рис. 16. Работа вкладки Объявления и push-уведомлений приложения IFTE-Info

Приложение в настоящее время находиться в режиме тестирования. Планируется размещение приложения в Google Play Market. Скачивание приложения будет доступно любому пользователю и будет осуществляться бесплатно. Единственным ограничивающим фактором является версия Android используемая на мобильном устройстве пользователя. Такое ограничение связано с тем, что приложение IFTE-Info использует для своего функционирования методы доступные в API 17 и выше, то есть версия Android у пользователя должна быть не ниже 4.1 Jelly Bean. Для скачивания приложения необходимо зайти в Google Play Market и в окне поиска ввести название приложения, либо фразы - теги (ургпу ифтэ, ифтэ, ургпу). Так же для привлечения внимания к приложению и удобства скачивания можно применить QR-код. QR-код – это двумерный штрих код, в котом может быть закодирована какая-либо информация. В данном случае в qr-коде будет заключена ссылка на скачивание приложения из Google Play Market. Разместить QR код можно на информационных стендах Института Физики, Технологий и Экономики УрГПУ. Студенты смогут при помощи своих Android устройств прочитать этот код и скачать приложение.

Во время создания приложение IFTE-Info неоднократно тестировалось и вносились изменения. Но для выявления всех недостатков необходимо массовое использование приложения. Так пользователь, установив и поработав с приложением, сможет оставить свой отзыв и пожелания в Google Play Market. На основе этих отзывов и будет основана дальнейшее развитие приложения.

К перспективам развития приложения можно отнести следующие пункты:

- повышение стабильности работы приложения;
- добавление новых функциональных возможностей (например раздел, несущий информацию о предстоящей сессии);
- перенос приложения на другие мобильные операционные системы (iOS и Windows Phone).

Заключение

Разработка приложений для операционной системы Android является актуальным вопросом в сфере информационных технологий. В ходе выполнения выпускной квалификационной работы была рассмотрена операционная система для мобильных устройств Android. Выявлены положительные и отрицательные стороны использования данной операционной системы. Были рассмотрены архитектура операционной системы Android и архитектура приложений работающих на ней. Произведен анализ инструментов программирования для Android. Поставленные задачи были выполнены:

1) анализ существующих решений;

2) анализ инструментов для программирования под операционную систему Android;

3) анализ возможностей программного обеспечения, языков программирования, библиотек для реализации приложения;

4) разработка приложения.

Цель поставленная в работе выполнена:

Разработано приложение, функционирующее на платформе Android, которое помогает студентам Института физики, технологии и экономики УрГПУ получать информацию о своем институте.

Список использованной литературы

1. Android 2. Программирование приложений для планшетных компьютеров и смартфонов. Рето Майер. – СПб.: Санкт-Петербург, 2011. – 672 с.

2. Android 4 для профессионалов. Создание приложений для планшетных компьютеров и смартфонов. Сатия Коматинени, Дэйв Маклин. – М.: Вильямс, 2012. - 880 с.

3. Android для программистов. Создаем приложения. П. Дейтел, Х. Дейтел, Э. Дейтел, М. Моргано. – СПб.: Питер, 2013. – 560 с.

4. Android для разработчиков. П. Дейтел, Х. Дейтел. - СПб.: Питер, 2016. - 512 с.

5. Android за 24 часа. Программирование приложений под операционную систему Google. Лорен Дэрси, Шейн Кондер. – М.: Рид Групп, 2012. – 464 с.

6. Android Ответы развития! [Электронный ресурс] // URL: https://commonsware.com/

7. Android. Программирование для профессионалов. Б. Харди, Б. Филлипс. - СПб.: Питер, 2016. - 640 с.

8. Android: разработка приложений для чайников. Фелкер Д. – М.: Диалектика, 2015. – 336 с.

9. Eclipse: разработка RCP-, Web-, Ajax- и Android - приложений на Java. Т. Машнин. – СПб.: БХВ-Петербург, 2013. - 384 с.

10. Google Android: системные компоненты и сетевые коммуникации. Голощапов А.Л. – СПб.: БХВ-Петербург, 2013. – 384 с.

11. Head First. Программирование для Android. Д. Гриффитс, Д. Гриффитс. - СПб.: Питер, 2016. - 704 с.

12. Java. Экспресс-курс [Электронный ресурс] // URL: http://developer.alexanderklimov.ru/android/java/java.php/

13. Блог на хабре о разработке под Android [Электронный ресурс] // URL: http://habrahabr.ru/blogs/android_development/

14. Введение в Android Studio. [Электронный ресурс] // URL: http://ds-release.ru/pervaya-programma-v-android-studio/

15. Виды приложений и их структура [Электронный ресурс] // URL: http://www.intuit.ru/studies/courses/12643/1191/lecture/21983

16. История операционной системы Android [Электронный ресурс] // URL: http://www.corporacia.ru/pages/page/show/2334.htm

17. Лекции по программированию под Android. [Электронный ресурс] // URL: http://www.slideshare.net/

18. Лекция Android. Fragments, ActionBar, Drawer [Электронный ресурс] // URL: http://www.slideshare.net/ssuserc84456/android-fragments-29111938

19. Операционная система Android корпорации Google [Электронный ресурс] // URL: http://bourabai.ru/os/android.htm

20. Освой программирование играючи [Электронный ресурс] // URL: http://developer.alexanderklimov.ru/

21. Официальная справка для Android разработчиков[Электронный ресурс] // URL: http://developer.android.com/index.html

22. Официальная справка по среде программирования [Электронный ресурс] // URL: http://www.jetbrains.com

23. Программирование для Android. Самоучитель / Денис Колисниченко . – СПб.: Санкт-Петербург, 2011. – 272 с.

24. Программирование под Android. Блэйк Мик . – СПб.: Санкт-Петербург, 2013. – 496 с.

25. Разработка игр под Android. Джером Ф. Димарцио. - СПб.: Питер, 2016. - 224 с.

26. Разработка приложений для Android. Хашими С., Коматинени С., Маклин Д. – СПб.: Питер, 2013. – 736 с.

27. Разработка приложений под Android. Основы Android. Курс лекций. [Электронный ресурс] // URL: http://www.udacity.com/

28. Смартфоны Android. Руководство пользователя / Андрей Жвалевский . – СПб.: Санкт-Петербург, 2014. – 224 с.

29. Создание приложений для Android за 24 часа. К. Делессио, Л. Дэрси, Ш. Кондер. – М.: Эксмо, 2015. – 528 с.

30. Статьи о программировании для Android [Электронный ресурс] // URL: http://flashbot.ru/android-dev

31. Уроки по основам языка программирования JAVA для начинающих [Электронный ресурс] // URL: http://www.fandroid.info/tutorial-po-osnovam-yazyka-programmirovaniya-java-dlya-nachinayushhih/

32. Уроки по основам языка программирования JAVA для начинающих [Электронный ресурс] // URL: http://www.fandroid.info/tutorial-po-osnovam-yazyka-programmirovaniya-java-dlya-nachinayushhih/

33. Философия Java, 4-ое издание. Эккель Б. - СПб.: Питер, 2012. - 638 с.

34. Форум о программировании для Android [Электронный ресурс] // URL: http://www.cyberforum.ru/android-dev/

35. Форум о программировании для мобильных устройств [Электронный ресурс] // URL: http://www.4pda.ru

36. Что такое платформа Eclipse и как ее использовать? [Электронный ресурс] // URL: https://www.ibm.com/developerworks/ru/library/os-eclipse/

37. Эффективное использование потоков в операционной системе Android. Технологии асинхронной обработки данных. А. Ёранссон. - М.: ДМК Пресс, 2015. 304 с.

Приложение 1

Пример кода выполняющего выбор курса, группы и отображение расписания:

```
spinner1.setPrompt("Группа...");
      spinner1.setVisibility(View.GONE);

          RadioGroup radiogroup = (RadioGroup)
getView().findViewById(R.id.radioGroup1);
          radiogroup.clearCheck();
              radiogroup.setOnCheckedChangeListener(new Radi-
oGroup.OnCheckedChangeListener() {

                    @Override
                    public void onCheckedChanged(RadioGroup group, int
checkedId) {

                          switch (checkedId) {

                          case R.id.radio0:
                                spinner1.setVisibility(View.VISIBLE);
                                imageView1.setVisibility(View.GONE);
                                break;
                          case R.id.radio1:
                                ArrayAdapter<?> adapter = ArrayAdapt-
er.createFromResource(getActivity(), R.array.twokurs, an-
droid.R.layout.simple_spinner_item);
                                adapt-
er.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
;
                                spinner1.setAdapter(adapter);
                                imageView1.setVisibility(View.GONE);
                                      spinner1.setVisibility(View.VISIBLE);
                                break;
case R.id.radio2:
      ArrayAdapter<?> adapter1 = ArrayAdapt-
er.createFromResource(getActivity(), R.array.threekurs, an-
droid.R.layout.simple_spinner_item);
      adapt-
er1.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_ite
m);
      spinner1.setAdapter(adapter1);
      imageView1.setVisibility(View.GONE);
            spinner1.setVisibility(View.VISIBLE);
```

```
                                    break;
case R.id.radio3:
        ArrayAdapter<?> adapter2 = ArrayAdapt-
er.createFromResource(getActivity(), R.array.fourkurs, an-
droid.R.layout.simple_spinner_item);
        adapt-
er2.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_ite
m);
        spinner1.setAdapter(adapter2);
                spinner1.setVisibility(View.VISIBLE);


                spinner1.setOnItemSelectedListener(new OnItemSelectedListener() {
                    @Override
                    public void onItemSelected(AdapterView<?> parent, View view,
                       int number, long id) {

                                switch (number) {
                      case 0:
                      imageView1.setVisibility(View.GONE);
                                    break;
                                case 1:
                                    imageView1.setVisibility(View.VISIBLE);
                                    im-
ageView1.setImageResource(R.drawable.img_1_1);
                                    break;
                                case 2:
                                    imageView1.setVisibility(View.GONE);
                                    break;
                                case 3:
                                    imageView1.setVisibility(View.VISIBLE);
                                    imageView1.setImageResource(R.drawable.zr);
                                     break;
                                }

                     }
                     @Override
                     public void onNothingSelected(AdapterView<?> arg0) {
                     }
                    });



        break;} }
```

Приложение 2

Пример кода выполняющего парсинг новостей с сайта УРГПУ:

```
class ParseText extends AsyncTask<String,Void,String> {


   @Override
   protected String doInBackground(String... params) {

        String str  = " ";

         try {

                Document document = Jsoup.connect(params[0]).get();
                      Element element = document.select(".itemFullText").first();
                      str = element.text();


   }catch (IOException e) {
                e.printStackTrace();}
         return str;
          }
   }

class ParseTitle extends AsyncTask<Void,Void,HashMap<String,String>> {


       @Override
       protected HashMap<String,String> doInBackground(Void... params) {
         HashMap<String,String>  hashMap = new HashMap<String, String>();

  try {        Document document =
Jsoup.connect("http://www.uspu.ru/novosti.html/").get();
                      Elements elements = document.select(".catItemTitle");
                      for (Element element : elements) {
                            Element element1 = element.select("a[href]").first();
                            hashMap.put(element.text(), element1.attr("abs:href"));
                      }
            } catch (IOException e) {
                  e.printStackTrace();
            }

            return hashMap; }}
```